Rapport de Stage Technicien ST2
Juin & Juillet 2009

Suivi GPS d'un drone par une antenne motorisée



Pierre-Alain Vercruysse,
Étudiant Génie Électrique 4°Année



Résumé :

En qualité d'étudiant en deuxième année de cycle ingénieur génie électrique, j'étais tenu d'effectuer un stage technicien. La difficulté cette année a concerné la recherche de ce stage. En effet, nous sommes nombreux à ne pas avoir trouvé de stage en entreprise. Cependant, une possibilité de stage a été trouvée à l'INSA même au sein du projet drone développé par celle-ci depuis maintenant plusieurs années.

Mon activité pendant ce stage concernait la station sol du drone. La station sol permet d'effectuer un suivi radar dont le rôle est de suivre le vol du drone grâce à la différence entre les coordonnées GPS de la station et celle du drone. Il fallait donc que je poursuive l’asservissement de cette antenne motorisée et que je propose de nouvelles cartes électroniques afin de modifier les défauts de la carte précédente.

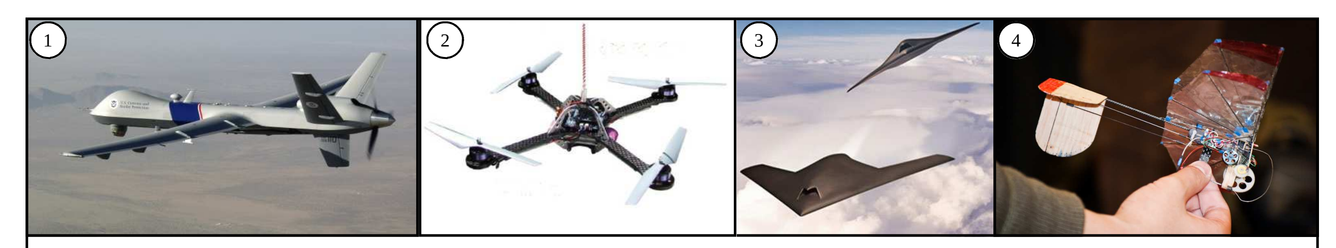
Concrètement, j'ai commencé par prendre en main la station sol existante. Il m'a fallu un certain temps d'adaptation. Après avoir bien intégré tout cet environnement de travail, j'ai donc commencé à modifier le code. Une fois le code modifié et performant, je suis passé au développement d'une seconde carte. Étant un peu en retard sur mon travail, je n'ai pas pu corriger les défauts pendant les deux mois du stage. J'ai cependant souhaité continuer à travailler sur la station sol pour pouvoir corriger les défauts majeurs. Il faudra malgré tout tester cette nouvelle carte pour la valider entièrement.

Sommaire

I - Le projet Drone

1) Les drones

Un drone est un véhicule volant non habité qui peut être autonome ou assisté par un pilote au sol. Le terme drone vient de l'anglais "bourdonnement" (commun au bruit de nombreux drone), et en anglais, un drone se dit UAV (*Unmanned Aerial Vehicle*). La plupart des drones que l'on retrouve actuellement trouvent leurs applications dans le domaine militaire. Ils ont pour but la surveillance et l'observation aérienne. Selon la nature de sa mission, un drone aura des dimensions, un fuselage, un poids bien particulier. À l'inverse de l'aviation ou de l'aéromodélisme, il n'y a donc pas de forme clairement définie pour un drone. Sur la Figure 1, on peut voir un ensemble de drones de tailles complètement différentes. On peut de manière générale classer les drones selon différentes caractéristiques tels que la taille, leur altitude de travail, et leur objectif.



1: *Le Drone "Predator" de l'armée américaine .*
2: *Quadrirotor (aéromodélisme) : X-ufo.*
3: *Le "Neuron" développé par la France et cinq autres pays européens (prévu pour 2010)*
4: *Drone miniature à ailes battantes présenté à l'EMAV 2009.*

Figure 1 : Différents types de drones

Historiquement, sans remonter à leurs balbutiements (lors de la première guerre mondiale), les drones ont été utilisés par les États-Unis lors de la guerre du Vietnam (1969) pour observer et détecter leurs ennemis. Cependant, c'est bel et bien depuis la guerre du golfe que ceux-ci font partie intégrante des armées modernes.

Comme dans beaucoup d'autres domaines quand il s'agit de haute technologies, les drones et l'armée sont étroitement liés. Cependant, on peut s'attendre à voir apparaître les drones dans le domaine civil. En effet, quelques applications types pourraient être la surveillance d'autoroutes (embouteillage, accidents, …), de forêts, de chemins de fer, des applications de topographie, etc…

2) Initiative du projet

À l'INSA Strasbourg, le projet drone commence dès 2003. Les débuts sont difficiles, et il faudra attendre une pré étude pour pouvoir commencer à travailler activement sur le projet. Au début une forme circulaire (soucoupe) avec une poussée par 4 moteurs avait été envisagée mais la forme a vite était repensée laissant place à une structure avec deux moteurs sur les extérieurs et une aile plane cylindrique (cf. Figure 2).

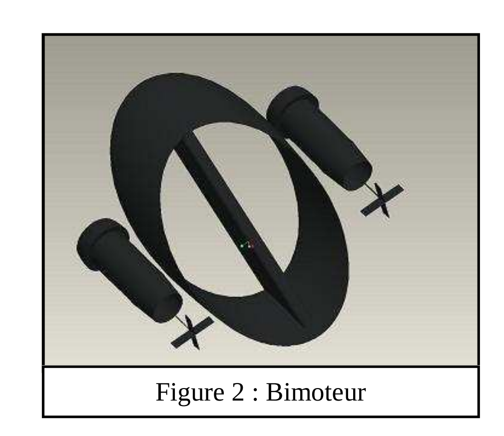


Figure 2 : Bimoteur

De manière méthodique, le premier prototype a été réalisé avec en fait un seul moteur thermique placé au cœur de la structure, et deux ailettes de direction sur les côtés (cf. Figure 3) .

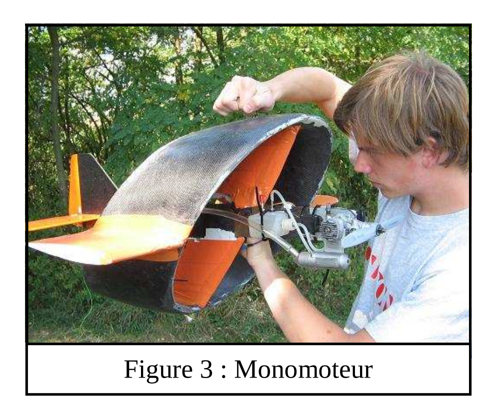
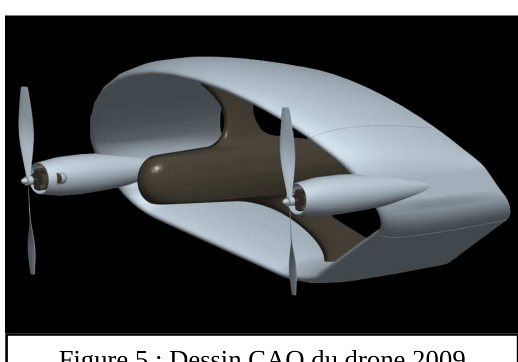

Figure 3 : Monomoteur


Figure 4 : Drone en vol en 2005

C'est après avoir stabilisé ce prototype que l'équipe du drone a pu participer au concours organisé par l'ONERA et la DGA en 2005. Ce premier concours avait pour objectif la surveillance de terrain au dessus d'un paysage urbain. Il verra d'ailleurs le drone Cigogne récompensé d'une première place ex-æquo accompagnée de la mention scénario pour avoir survolé la zone et identifié des obstacles. La photographie en Figure 4 montre le drone version 2005 en vol.

Actuellement le drone a une forme cylindrique avec deux moteurs brushless situés sur ses flancs. Les Figures 5 et 6 montrent le drone dans cette configuration. Il est également possible de retrouver une vidéo du drone en vol en suivant le lien suivant :

http://www.insa-strasbourg.fr/medias/dossier-drone/vol--horizontal-drone-cigogne.wmv


Figure 5 : Dessin CAO du drone 2009

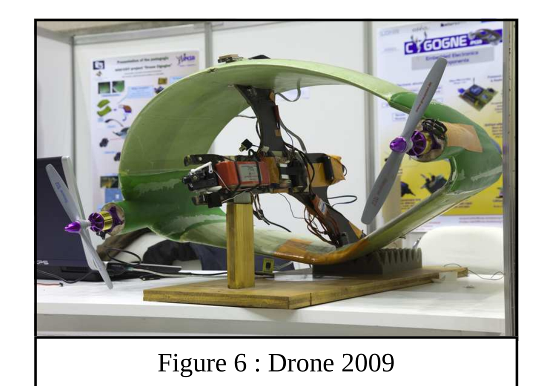


Figure 6 : Drone 2009

3) Organisation du projet

Le projet est géré par un trio de professeurs : Renaud KIEFER, Marc VEDRINES, François KIEFER. C'est un projet développé dans la section mécatronique de l'INSA. Cependant, le projet étant pluridisciplinaire, des étudiants d'autres spécialités sont également mis à contribution : Génie Electrique, Plasturgie, Mécanique, Topographie, etc…

Le développement du drone est rendu possible grâce aux laboratoires des différentes spécialités de l'école. La totalité du drone est réalisée au sein de l'INSA : moulage, pièce mécaniques, développement des cartes électronique, traitement des images, …

Le projet s'articule autour de quatre grands groupes de travail :
- L'équipe *Interface Homme Machine* qui est chargée du développement de toute l'électronique embarquée sur le drone. Cela comprend les parties acquisition, traitement, et transmission des données.
- L'équipe *Mécanique* chargée de la conception et la fabrication de la structure du drone.

- L’équipe *Vision* s’occupant du test et de la réalisation des différents prototypes de systèmes de vision embarquée.
- L'équipe *Station Sol* réalise les infrastructures nécessaires au contrôle et au suivi du vol. Cela comprend une interface graphique ainsi qu'une infrastructure de lancement du drone et un système de suivi de trajectoire de l'antenne au sol.

Pour ma part, j’ai donc travaillé dans l’équipe station sol. J’ai repris le projet des étudiants mécatroniciens précédents. Je me suis occupé en particulier de l’amélioration de la station sol (code et carte électronique), ainsi que de la mise en place d’une communication série avec un ordinateur. Pour ce faire, j’ai travaillé avec deux camarades de classe Gaylord Wagner (interface Labview sur PC), et Emmanuel Roussel (Développement d’un Joystick commande de haut niveau, développement d’un compas, …). Au total, nous étions 7 élèves électroniciens à travailler en stage sur ce projet.

4) Détails des différents groupes de travail

4) a - Groupe électronique embarquée

Tout d’abord, il y a le groupe d’électronique embarquée (IHM). La carte mère permettant de contrôler le drone est entouré d’un grand nombre de périphériques (carte capteur, GPS, Centrale Inertielle, Radio Commande, émetteur Xbee, …). Sur la Figure 7 est représentée l’architecture de cette carte.

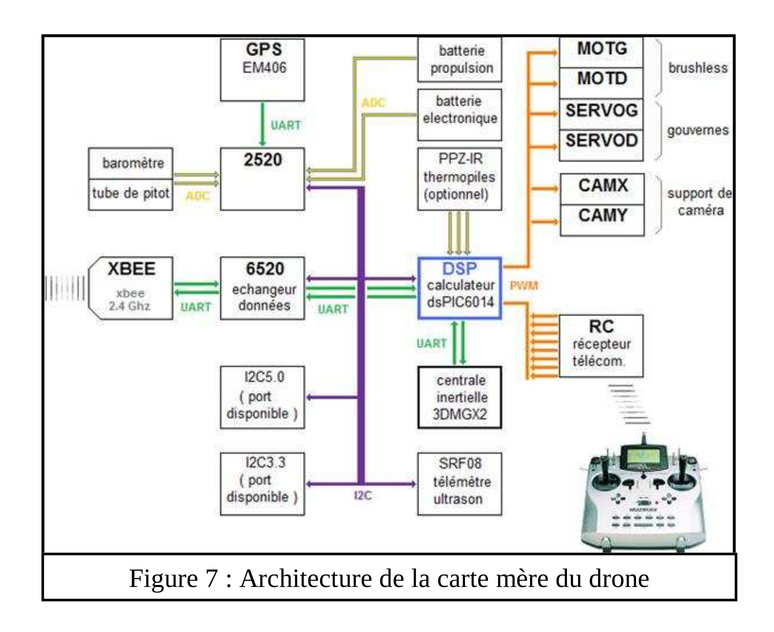


Figure 7 : Architecture de la carte mère du drone

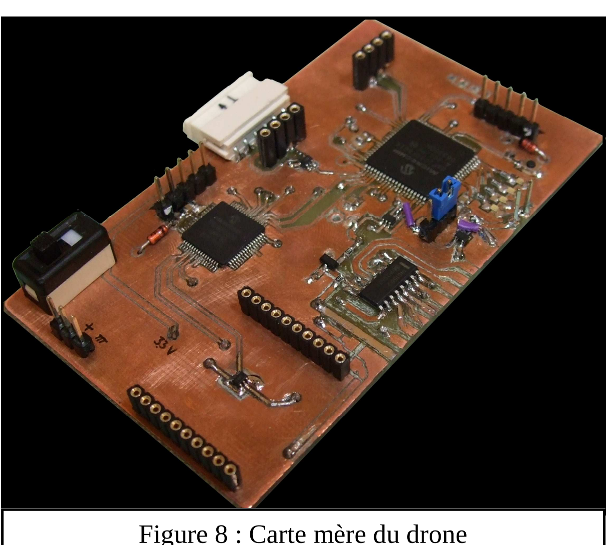


Figure 8 : Carte mère du drone

La grosse partie du travail dans ce groupe de travail est la programmation des microcontrôleurs gérant la commande des moteurs et des servo-moteurs (asservissement) en fonction des données reçues provenant de l’ensemble des capteurs et de la Radiocommande. Le groupe de travail a connu un grand nombre de participants dont notamment deux élèves électroniciens en par. Le travail sur cette carte sera prolongé en PRT avec certainement une modification (simplification) de la carte pour permettre une gestion plus simple de la commande du drone.

4) b - Groupe Mécanique :

La structure du drone est composée de 5 éléments assemblés (aile supérieure, aile inférieure, 2 ailes latérales et mât central) et rigidifiés par l'apport de longerons et de polystyrène expansé. Les ailes supérieure et inférieure sont en structure « sandwich » alors que les ailes latérales sont exclusivement en composite. Ces éléments sont le fruit d'un processus de fabrication très précis.

Après la phase de conception CAO/FAO, la première étape nécessite l'usinage de bruts, généralement en bois, permettant de réaliser les moules par la suite. Il s'agit des masters extrados et intrados (cf. Figure 9). On applique ensuite du "gel coat", améliorant l'état de surface des masters et augmentant la durée de vie du moule en durcissant la surface externe.

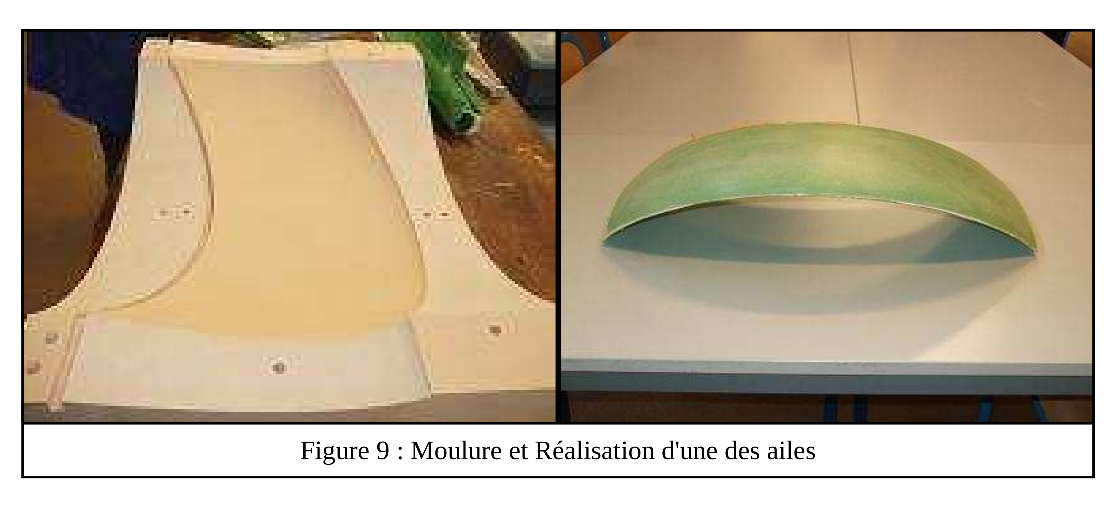


Figure 9 : Moulure et Réalisation d'une des ailes

Une fois les masters finalisés, on procède à la seconde étape, la création des moules (intrados et extrados). Ces moules en composite son réalisés par moulage à partir des masters. Leur composition leur procure une durée de vie importante. Il est donc possible de fabriquer un grand nombre d'ailes, assez rapidement.

Enfin, il est possible de réaliser les ailes proprement dites. Pour cela, chaque surface externe des ailes est d'abord conçue indépendamment par l'enduit de résine époxy placé sous vide pendant 24 heures. Les deux parties sont ensuite assemblées et collées pour former l'aile finale.

4) c - Groupe Vision :

Le groupe vision se concentre sur deux aspects : la capture de vidéos à bord du drone et l'émission de cette vidéo en direct soit sur un ordinateur soit sur des lunettes de vision 3D (cf. Figure 10). Les lunettes 3D quant à elles impriment la vidéo accompagné éventuellement d'une incrustation d'information capteurs tel que la vitesse, l'altitude, etc…



Figure 10 : Lunettes 3D

Concernant la caméra elle est orientable dans toutes les directions grâce à deux servo-moteurs. Cet ensemble est monté à l'avant du drone comme on peut le voir sur la Figure 11.

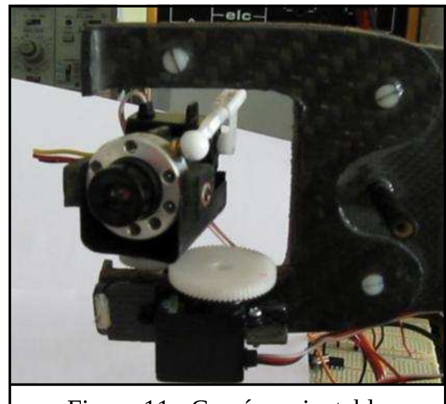


Figure 11 : Caméra orientable

4) d - Groupe Station sol :

Il s'agit du groupe que j'ai intégré. Ce groupe s'occupe logiquement de la mécanique de la station ainsi que de l'électronique qui va permettre de communiquer avec le drone pour lui fournir des consignes. La station (cf. Figure 12) sol permet de diriger des informations ainsi que de faire un suivi d'antenne (cf. Figure 13).

À ces deux activités, s'ajoute également l'activité topographique qui s'occupe de réceptionner les images issues de la caméra et de les traiter à des fins topographiques.

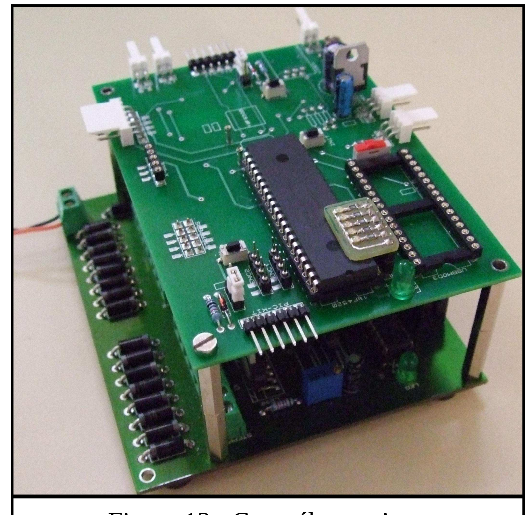


Figure 12 : Carte électronique

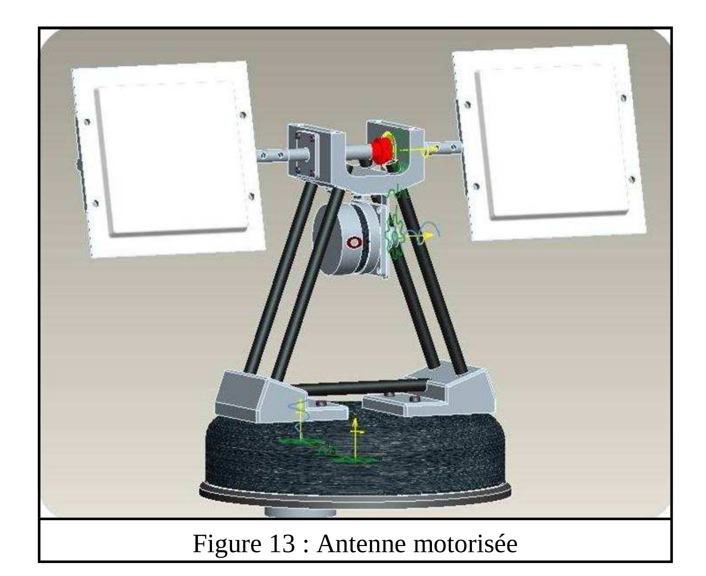


Figure 13 : Antenne motorisée

II - Organisation du travail pendant le stage

1) Diagramme de Gantt

Afin d'y voir plus clair et de nous fixer des limites, il nous a été demandé de construire un diagramme de Gant. Bien entendu il y a un décalage entre le diagramme de gant prévisionnel et le déroulement effectif du stage. Le diagramme suivant indique le diagramme prévisionnel ainsi que les activités effectuées.

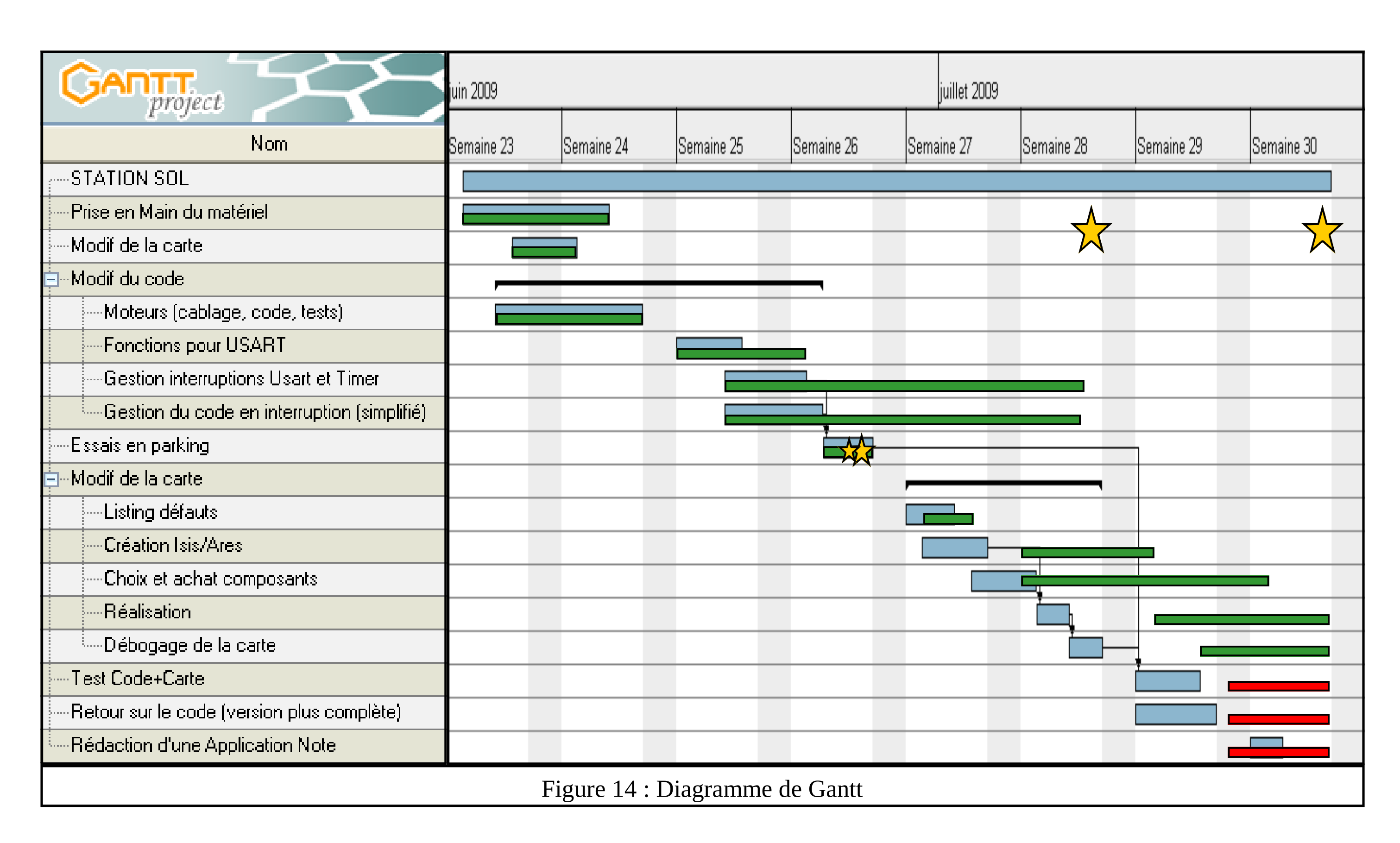


Figure 14 : Diagramme de Gantt

La prise en main du matériel a été assez rapide. Cependant, on remarque sur ce diagramme que la gestion du code a été plus complexe que ce qui avait été imaginé au début. Les quelques essais (tests parking et tests sur toits) ont été assez concluants. Suite à la validation par ces essais, le travail s'est donc porté sur la réalisation d'une nouvelle carte. Pour ce faire, il fallait exactement définir ce que l'on voulait ajouter à cette nouvelle version. Il était assez délicat au début de savoir exactement ce qui devait être amélioré. De nombreuses consultations avec mes camarades ont été très utiles pour définir une station sol qui pourrait être la plus efficace possible. Par exemple, l'ajout d'un Xbee avec la possibilité d'une configuration *in-circuit* a été la source d'un grand nombre d'erreurs difficilement décelables.

La plus grande erreur visible sur ce diagramme est la mauvaise synchronisation lors de l'achat des composants : la liste des composants était disponible assez rapidement (Semaine 28). Cependant le retard dans la livraison de la commande (envoi de la liste puis réception) est responsable du retard qui apparaît sur cette tâche. Par ailleurs à cause d'une erreur de conception de la carte (Adaptation des niveaux 3.3V → 5V du Xbee), ce retard s'est amplifié.

De manière à réparer cette erreur sur la carte et pour permettre une meilleure prise en main future de la station sol, le travail de débogage que j'ai commencé sur cette carte s'est poursuivi

jusqu’à fin septembre. Les problèmes ont été corrigés et consignés sur une note d’application.

Finalement, certaines tâches n'ont pas pu être validées : le code sur la station sol V2 n'a pas pu être testé ce qui est fort regrettable. Une amélioration du code n'a pas pu être pensée. Cependant, le code fonctionne assez bien tel quel. Pour finir la rédaction de la note d'application n'a pu se faire que en dehors du stage.

2) Archivage des fichiers : SVN

Pour permettre un travail plus efficace, plus sécurisé et avec plus de suivi, un SVN a été utilisé. Un SVN est en fait un Système qui permet de faire des sauvegardes du travail en cours sur un site internet. Ceci est très pratique en développement de projets car cela permet de ne pas perdre le travail effectué lorsque l’on souhaite modifier, ajouter ou supprimer certaines choses. Le SVN fait en fait des sauvegarde sur internet. Selon les cas, il est possible d’accéder à l’ensemble des dossiers du projet. C’est donc également très pratique pour l’échange de documents au sein même du groupe de projet. Pour utiliser un SVN il faut appartenir à un « groupe de travail » (une simple adresse e-mail suffit pour appartenir à ce groupe). Dans notre cas, le SVN s’appuyait sur les outils de Google. Un compte Google (enregistrement possible avec une adresse du type …@insa-strasbourg.fr) est donc nécessaire.

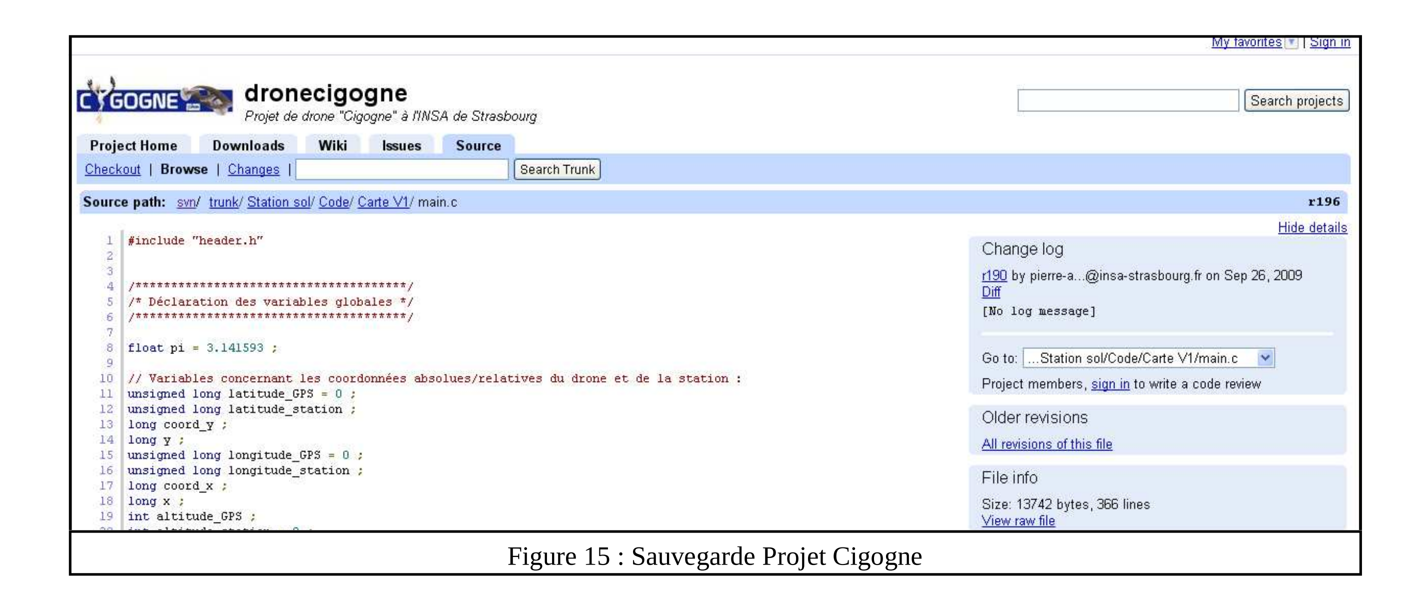


Figure 15 : Sauvegarde Projet Cigogne

Un accès libre au SVN est possible en suivant le lien suivant : *http://code.google.com/p/dronecigogne/*. La Figure 15 représente ce que pourrait voir un utilisateur libre. Il est également possible pour lui de télécharger les fichiers si il a au préalable installé un logiciel gérant les SVN sur son ordinateur.

L’utilisation d’un SVN ne modifie en rien l’organisation des fichiers sur le système d’exploitation. Il y a juste l’ajout de quelques options lors d’un clique droit sur un fichier ou dossier contenus dans le SVN. Sur la Figure 16 on peut voir par exemple qu’il est possible de mettre à jour un dossier/fichier en cliquant sur "SVN Update" (récupère le fichier d’origine stocké sur internet), de publier la nouvelle version d'un fichier ou de rajouter un nouveau dossier/fichier en cliquant sur "SVN Commit". Il est également possible d'accéder à de nombreuses autres possibilités : supprimer un fichier du SVN, interdire l'ajout d'un fichier sur le SVN, etc…

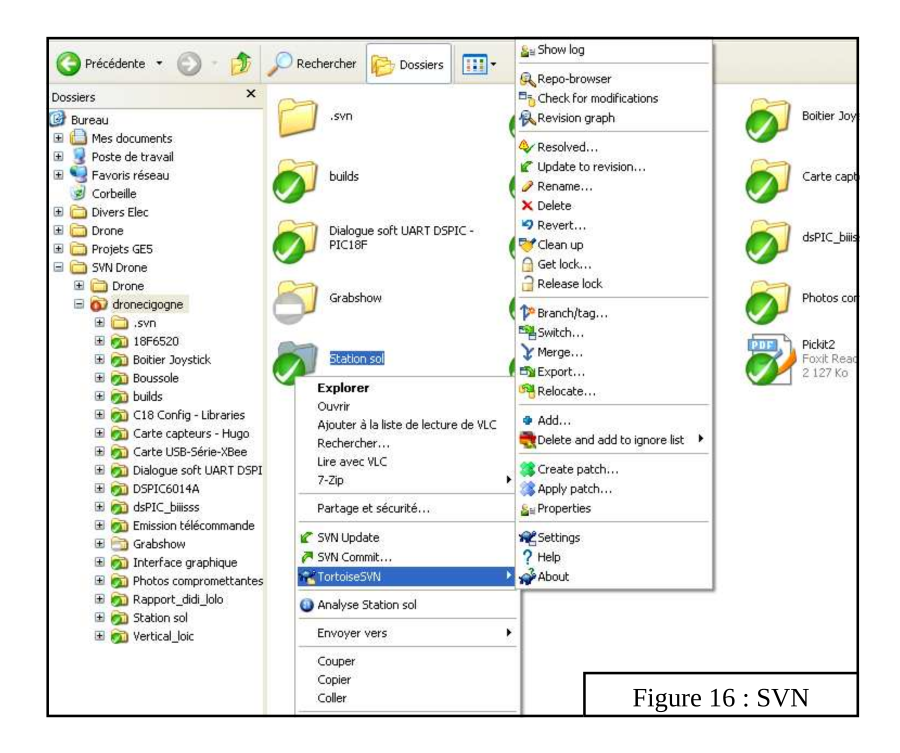


Figure 16 : SVN

3) Travail en équipe

Personnellement, je travaillais sur un projet assez indépendant puisque la plus grande interaction que j'avais était celle avec le responsable de la carte capteur laquelle devait m'envoyer une trame sous un certain format. Malgré cela, je me suis rendu compte de l'importance de connaitre un projet dans sa globalité pour pouvoir travailler efficacement dessus. En effet, il me fallait également certaines informations concernant les protocoles pour interfacer correctement le Joystick, le PC s'occupant du suivi du drone, etc…


Figure 17 : Quelques réglages lors des essais

D'un point de vue plus humain, il a été très agréable de travailler dans une équipe dans laquelle régnait une très bonne ambiance. Le fait que nous nous connaissions tous déjà a contribué justement à cette bonne ambiance.



Figure 18 : Premiers tests sur le toit du Bâtiment C

4) Phases de Tests

Toutes les phases de tests ne se sont en fait pas déroulés sur la maquette du drone 2009 (Figure 6) mais plutôt sur un "mullet" (Figure 17) qui est en fait un petit avion plus stable que le drone permettant de valider un grand nombre de travaux effectués.

Avec ce projet, je me suis également rendu compte de la très grande importance de prévoir un maximum de tests. Il m’a été possible d’effectuer quelques tests sur les toits de l’INSA pour vérifier le bon asservissement de l’antenne. En effet, travailler sur le toit permet de contourner les problèmes de réception du GPS observés lors des tests sur le parking de l’INSA. Ces tests ont permis de corriger les premiers défauts avant les essais sur le terrain.

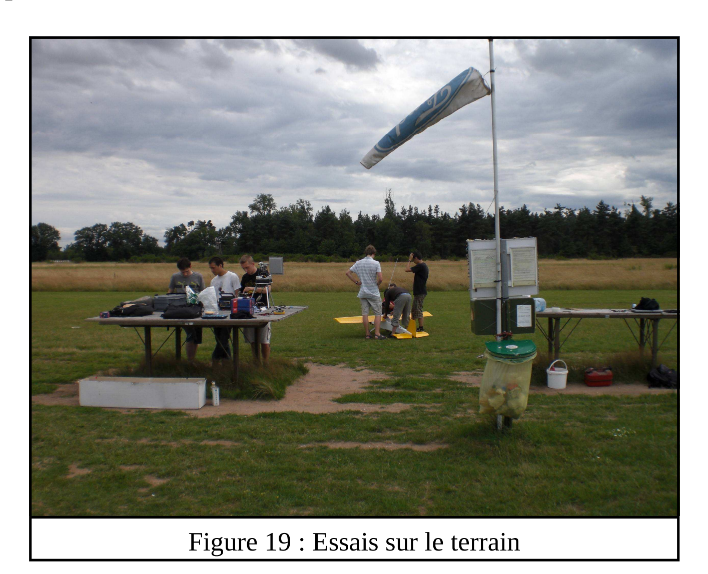


Figure 19 : Essais sur le terrain

Concernant les essais sur le terrain, deux ont été effectués. Malheureusement, je n’ai pu être présent entièrement lors du premier essai et étant donné les complications qu’il y a eu durant les premières heures de cet essai (en particulier problèmes avec la carte capteur et le GPS), je n’ai pas eu le temps de tester personnellement quoique ce soit avant mon départ. Les deuxièmes essais se sont déroulés le dernier jour du stage, et ne se sont une nouvelle fois pas déroulés comme prévu. En effet, quelques problèmes sur la carte mère sont survenus au début, et une fois ces problèmes

rectifiés, les batteries embarquées n'étaient finalement plus très fiables et par conséquent, l'antenne ne pouvait plus suivre de manière efficace le drone puisque la puissance d'émission du Xbee était moindre.

Par ailleurs, nous nous sommes rendus compte de perturbations dues au moteur thermique utilisé puisque la transmission fonctionnait assez bien lorsque le moteur était coupé (vol plané).

Au final, je me suis rendu compte de l'importance qu'il fallait accorder à la préparation des tests ainsi que du listing du matériel à amener sur place. Par exemple, pour un essai intermédiaire, j'avais oublié de prendre avec moi le composant principal de la station sol.

5) Présentation publique du projet

5) a - Journées Portes Ouvertes à l'aérodrome du polygone

Le deuxième week-end du stage (13 et 14 juin), il nous a été proposé de présenter le drone lors de la journée portes ouvertes de l'aérodrome du Polygone (Neuhof). Cette journée avoir pour but la publicité de cet aérodrome, et donc au passage un ensemble de stands étaient présents (Armée, Drone Cigogne, club d'aéromodélisme, etc…) pour présenter leurs activités. C'était également l'occasion d'un petit meeting aérien puisque on pouvait y voir d'anciens avions de la seconde guerre mondiale par exemple.



Figure 20 : Antenne motorisée

Cette journée a été très intéressante puisque non seulement elle nous a appris beaucoup de choses sur le projet drone en lui-même, mais elle nous a aussi permis d'expliquer à des visiteurs (sans nécessairement avec un bagage scientifique) ce projet. De nombreux étudiants étaient présents lors de cette journée, et je pense qu'il serait très intéressant de continuer à assister aux prochaines portes ouvertes de ce type car il s'agit là d'un moyen efficace de plébisciter le projet et l'école.

5) b - Conférence EMAV 2009 (Pays-Bas)

L'EMAV est la conférence Européenne sur les drones (Micro Air Vehicle). Chaque année, elle se déroule dans une ville différente. Cette conférence permet de réunir professionnels, étudiants et tous ceux concernés par des projets type drone. Cette année, l'EMAV se déroulait à Delft (Pays-Bas) et était organisée par l'Université TU Delft. Le contenu de ces quatre journées est assez divers : présentation publique des projets de chaque équipe, essais "outdoors" et "indoors". Du côté des équipes, on retrouve principalement des équipes coréennes et françaises (beaucoup d'écoles d'aéronautique et aviation). Côté professionnel, on retrouve l'ONERA, la DGA, Représentants de l'armée américaine, etc…



Figure 21 : Présentation lors de la conférence

Les essais "outdoors" : pour ces essais un certain nombre de points était attribué aux groupes qui arrivaient à faire un atterrissage précis, un largage, une crevaison de ballon, une localisation de véhicule et/ou le passage sous une arche. Des bonus étaient accordés aux équipes qui réalisaient ces missions en autonome ou par guidage aux lunettes.



Figure 22 : Essais "outdoors"

hit balloon
fly arch trough arch
locate vehicle
count number of persons around vehicle, and identify uniform color
search zone
take-off
precision landing
regular landing
paintball drop, choose zone based on color of uniforms

Figure 23 : Mission "outdoors"

Les essais "indoors" se déroulaient en deux parties. Une des deux missions était d'entrer dans une pièce, de récupérer un objet posé sur une table, de le déposer sur une zone spéciale et d'atterrir sur une plateforme prévue.

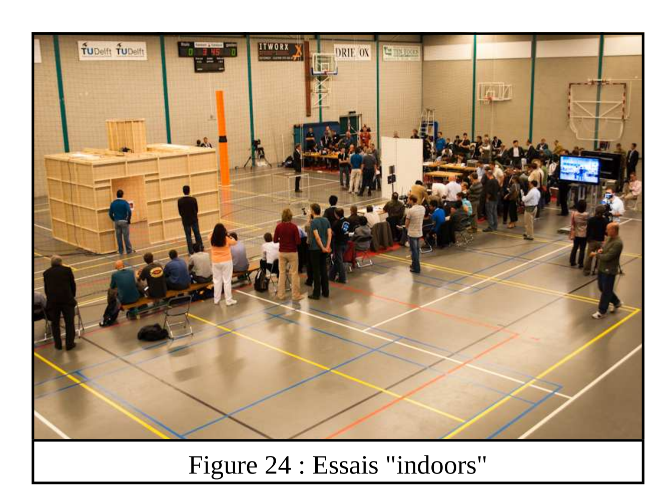


Figure 24 : Essais "indoors"

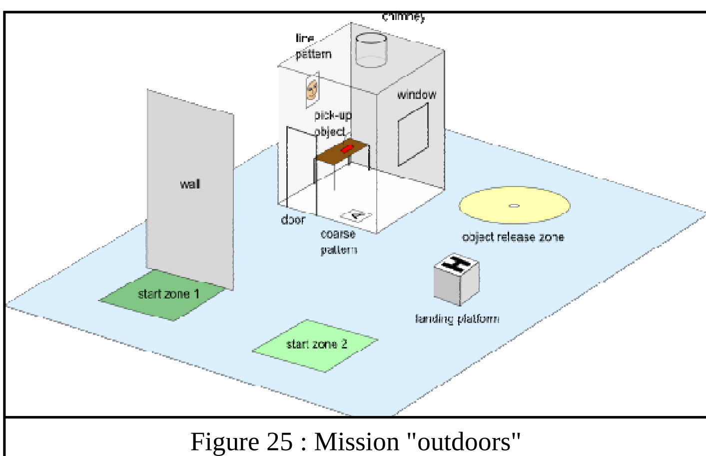


Figure 25 : Mission "outdoors"

Pour plus d'informations sur cette conférence, il est possible de visiter le site en suivant le lien suivant *http://www.emav2009.org/*

III - Descriptif de la station sol :



Figure 26 : Antenne motorisée

1) Rôle de la station sol :

L'objectif principal de la station sol est de faire un suivi du drone par une antenne (*cf. Figure 36*). Le drone étant équipé d'un émetteur vidéo, la station sol récupère le flux vidéo par un module de réception posé sur le mat, et retransmet ce flux directement par USB grâce au module "Grabshow". Par la suite, il serait intéressant d'intégrer une seconde antenne qui ferait la réception et l'émission des trames Xbee Sol-Air d'abord puis par exemple Sol-Sol (pour une télécommande ou le Joystick). La station sol a par ailleurs d'autres rôles. Voici un listing de ses taches :

- Acquisition de la trame GPS (@#) du drone pour effectuer le suivi d'antenne.
- Acquisition de la trame GPS du drone pour un ordinateur branché sur la station sol (USB).
- Acquisition de la trame PID (@P) de l'ordinateur (USB) afin de modifier les PID en vol.
- Acquisition des consignes Joystick (en mode filaire actuellement).
- Envoi des consignes Joystick et de la trame PID.
- Envoi de certaines données à afficher sur les lunette 3D.

Une première station avait été développée lors d'un projet MIQ4 en 2008/2009 (*cf. Figure 37*). Ce travail avait débouché sur la réalisation d'une double carte commande et puissance. Cette carte qui est tout à fait fonctionnelle présente cependant quelques défauts. Une nouvelle version de ces cartes a donc été réalisée.

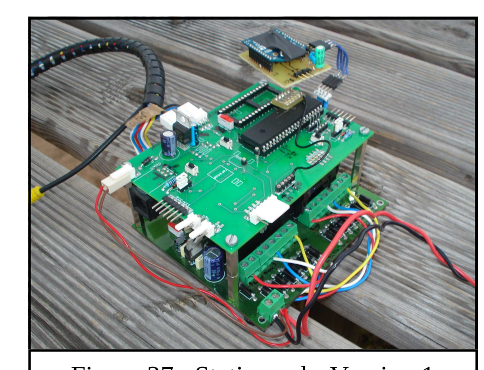


Figure 27 : Station sol - Version 1

2) Schéma du drone en vol :

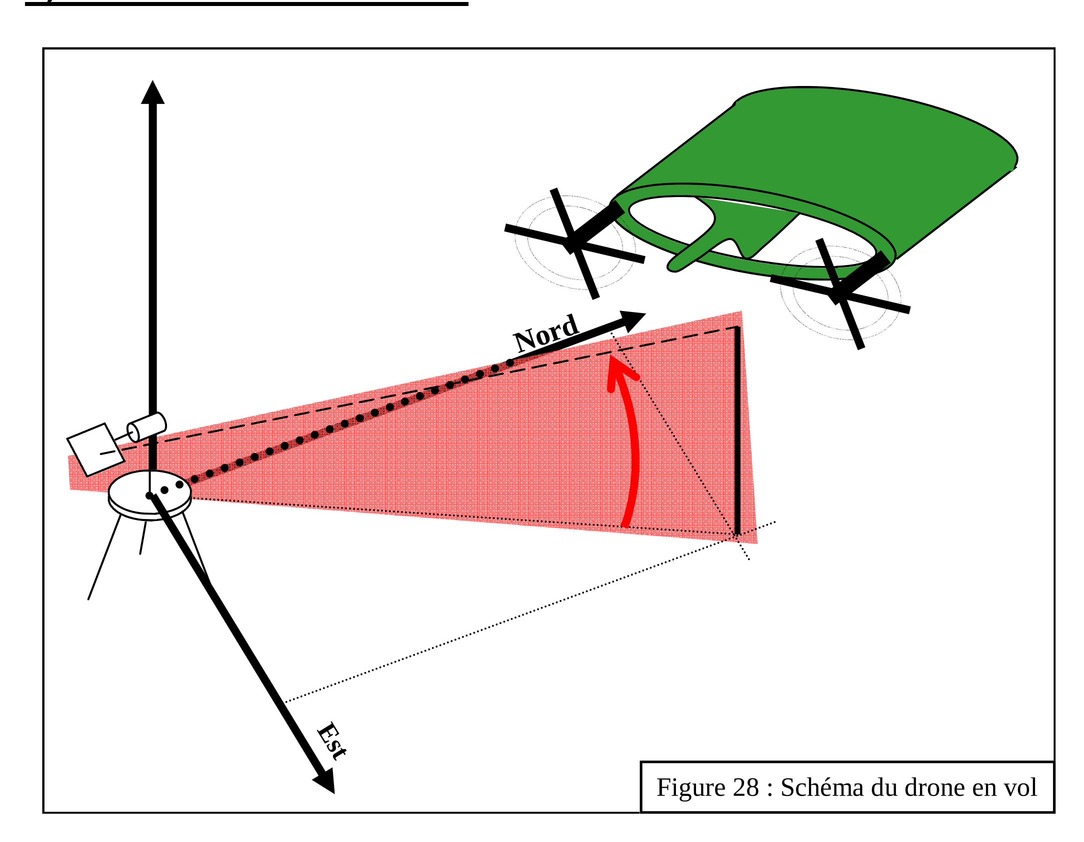


Figure 28 : Schéma du drone en vol

3) Diagramme de fonctionnement :

La carte s'appuie sur 2 microcontrôleurs PICs 18F. L'un va se charger de motoriser l'antenne afin d'orienter celle-ci toujours dans la direction du drone. Le second PIC va être un aiguilleur d'information pour l'incrustation vidéo et l'envoi de la trame pour la correction des PIDs.

Le Xbee transmet la trame GPS aux deux PICs ainsi qu'au PC (interfacé en USB) contenant l'interface graphique. Le PC va ensuite renvoyer une trame PID contenant les corrections à apporter en vol sur les PID, ainsi que les données qu'il faudra afficher lors de l'incrustation vidéo. Le PIC aiguilleur va donc récupérer la trame GPS ainsi que la trame PIDs. Il va par ailleurs recevoir des consignes Joystick. À partir de ce flux d'informations, il va construire la trame PID (@P) envoyée par Xbee au drone. Il va également afficher sur les lunettes 3D les informations souhaitées (Altitude, Vitesse …) par le biais du module de surimpression vidéo (incrustation vidéo). Le switch SW permet de reconfigurer le Xbee par USB lors d'un déréglage de celui-ci.

Voici un diagramme de fonctionnement de la station :

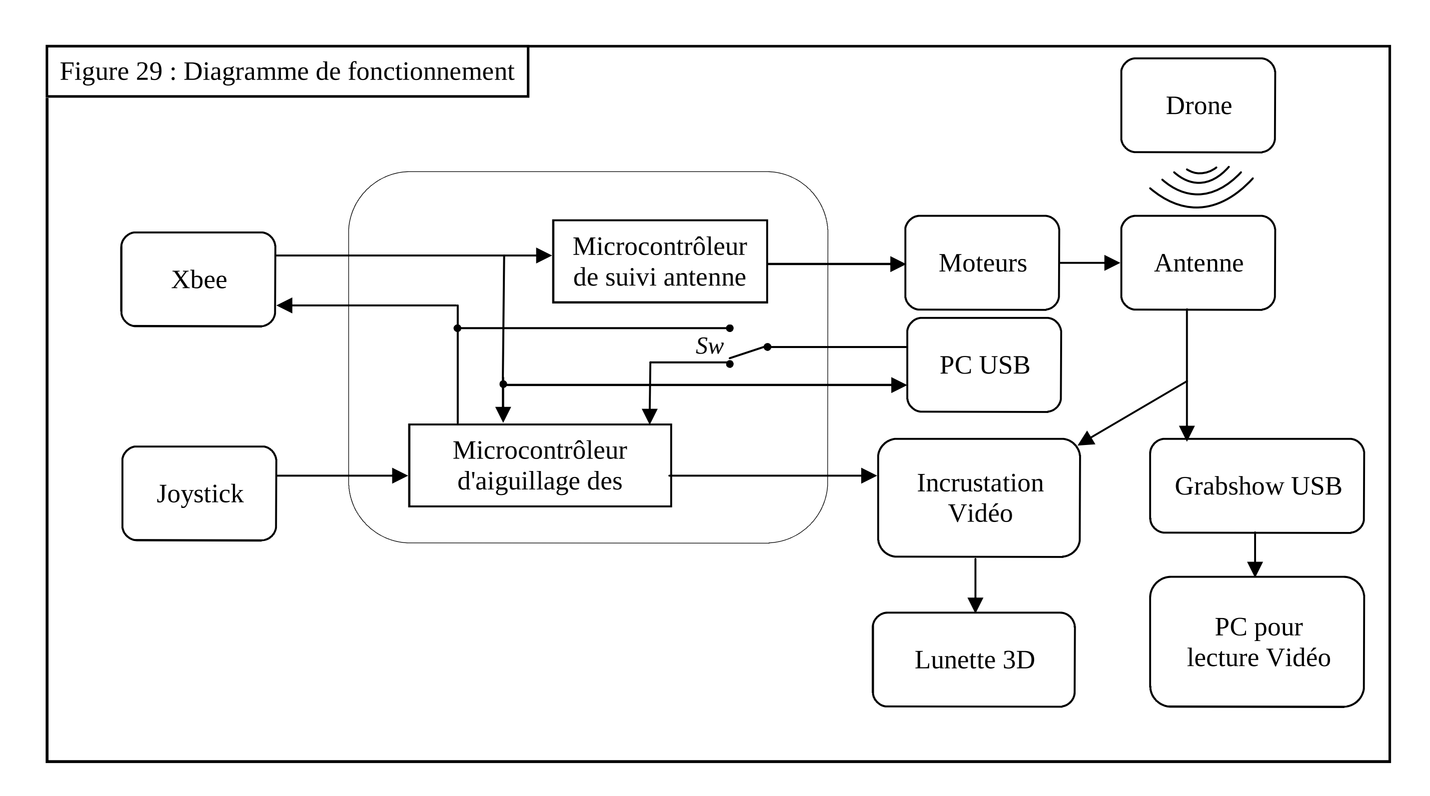


Figure 29 : Diagramme de fonctionnement



Figure 30 : Récepteur Vidéo et Grabshow USB

4) Coordonnées GPS :

Les coordonnées GPS sont envoyées par la trame GPS (@#). Elles ont pour unité les degrés et se composent de la longitude (axe "X") et de la latitude (axe "Y") ayant pour origine respectivement le méridien de Greenwich et l'équateur. La Terre étant approximée à une sphère de rayon R = 6374 km, par un simple produit en croix, on peut trouver une équivalence entre les degrés et les mètres : 360° = 2π.R.

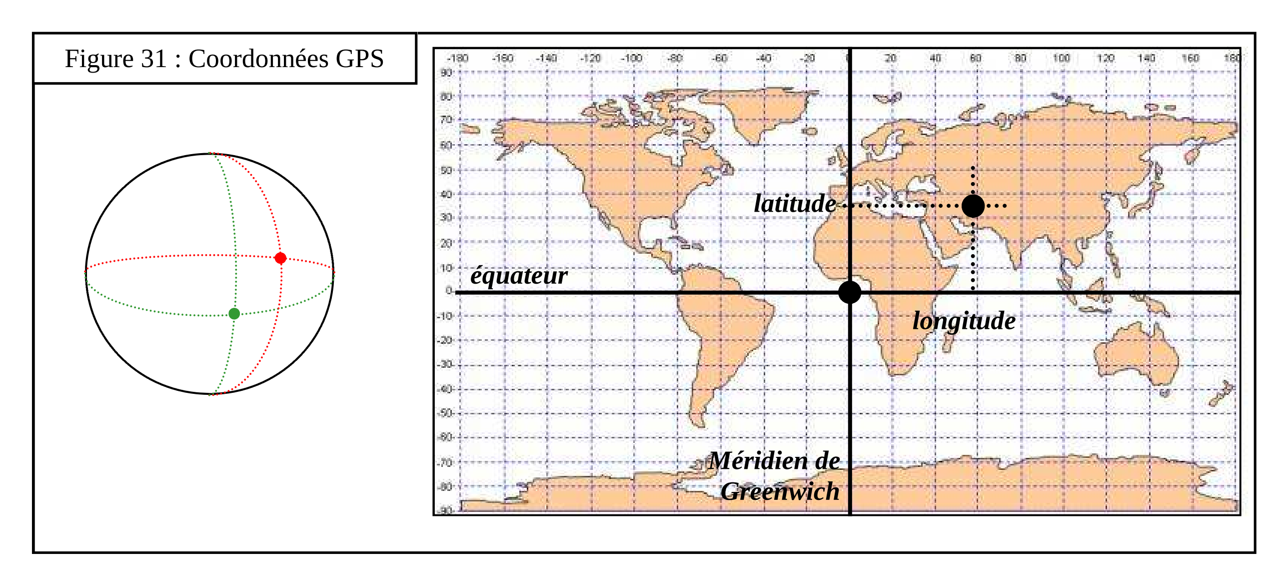


Figure 31 : Coordonnées GPS



Figure 32 : Impression écran issu du site "gpsfrance.net"

IV - Descriptif de la carte électronique :

1) Contexte de la station sol :

La carte électronique s'inspire logiquement de l'architecture souhaitée (Figure 29). La structure de la carte suit celle de la précédente version. La station sol s'articule autour de deux microcontrôleurs PIC : un 18F4520 pour la gestion de la commande des moteurs, et un 18F6520 qui s'occupera de l'aiguillage des données entre le Xbee, le PC, les lunettes 3D ainsi que le Joystick.

1) a - Choix des microcontrôleurs :

Les PICs retenus pour cette deuxième version de la carte restent les mêmes que pour la première version. Il faut cependant noter que le PIC 18F4520 est sans doute surdimensionné. En effet, un grand nombre d'entrées/sorties restent inutilisées. Un PIC comme le 18F2221 (28 pins au lieu de 40) pourrait remplir ce rôle. Cependant, pour éviter de s'éloigner de la carte V1, le même PIC a été gardé. En ce qui concerne le PIC échangeur (18F6520), il a été choisi pour ses deux ports UART. Ce PIC a été parfaitement choisi puisque seuls deux PICs possèdent 2 UART (communication série) et 4 CCP (Capture Compare). Cependant, ce PIC n'existe que au format CMS TQFP(64pins), ce qui est un peu embêtant pour les soudures non professionnelles.

1) b - Organisation de la carte :

Voici un schéma simplifié de l'organisation de la station sol :

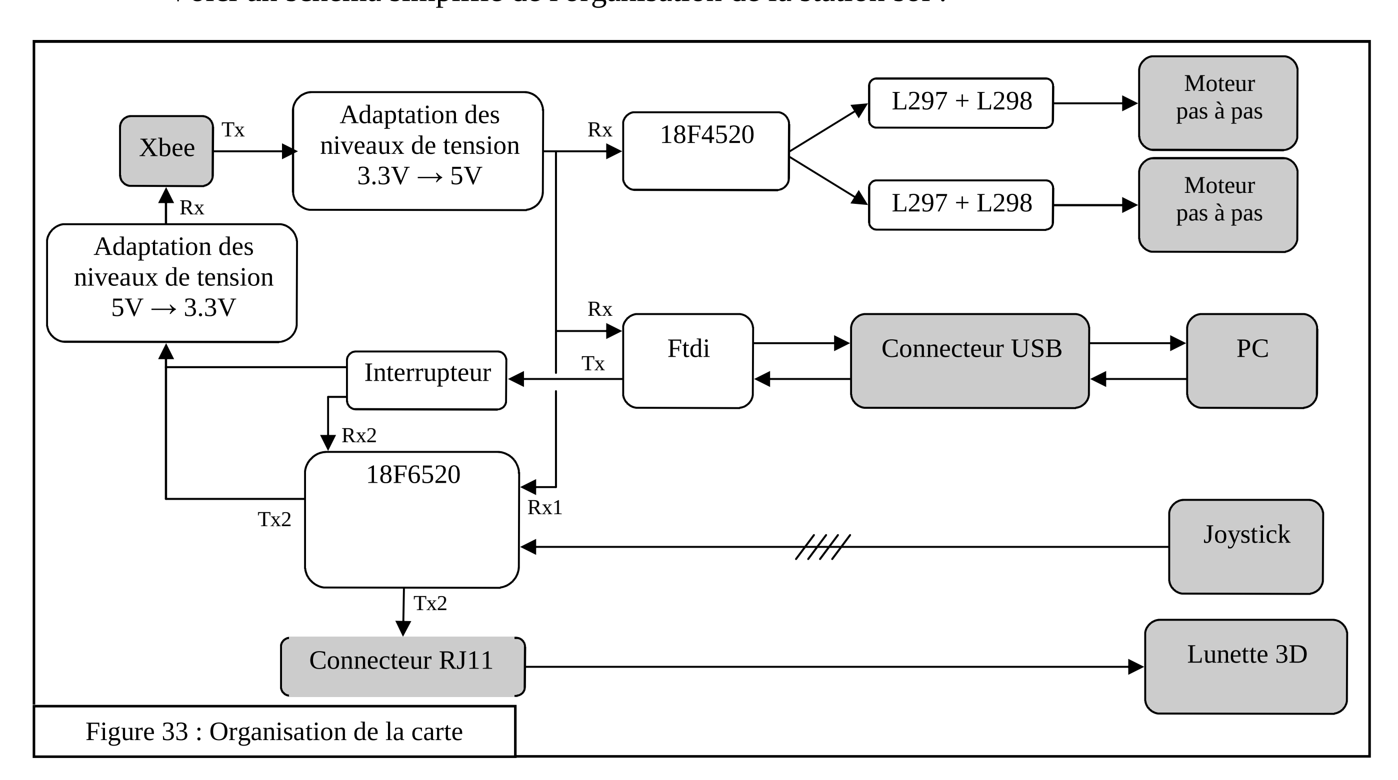


Figure 33 : Organisation de la carte

Ce schéma ne fait intervenir que les composants les plus importants. À ces composants s'ajoutent des réseaux de 6 LEDs (4 vertes, 2 rouges) pour chaque PIC ainsi que des Interrupteurs, un Bouton Poussoir, des connecteurs PicKit, un connecteur pour le Compas (Projet annexe), etc…

2) Détails des différents composants de la station sol :

2) a - Cablage du FTDI :

Concernant le FTDI : il s'agit d'un composant permettant la conversion de signaux UART en signaux USB directement lisibles par un ordinateur grâce à un logiciel comme X-CTU. L'ordinateur (qui est en fait l'interface graphique) récupère donc les informations de la trame GPS (coordonnées, altitude, vitesse, etc…), et renvoie une trame PID (corrections des PID, sélection des waypoints, affichage d'informations sur l'incrustation vidéo …) par le biais du 18F6520. Ce dernier, le PIC d'aiguillage va récupérer les consignes Joystick, les PID et va les retransmettre par Xbee de façon structurée. Il a également été convenu que le PC enverrai une première trame indiquant quelles indications devaient être imprimées sur les lunettes (cf. explication sur le code de ce PIC). Le pic d'aiguillage se chargera donc d'envoyer régulièrement les informations à afficher sur la lunette.

Le câblage de ce composant est indiqué sur le schéma ci-dessous. Les diodes D17 et D18 permettent d'éviter un conflit entre les alimentations de l'USB et celle de la carte. En effet, si ces diodes n'étaient pas là, il y aurait le 5V USB qui irait directement sur le 5V régulé de la batterie. Dans tous les cas, le FTDI sera alimenté par une seule des deux tensions (la plus grande), et les deux alimentations resteront séparées l'une de l'autre.

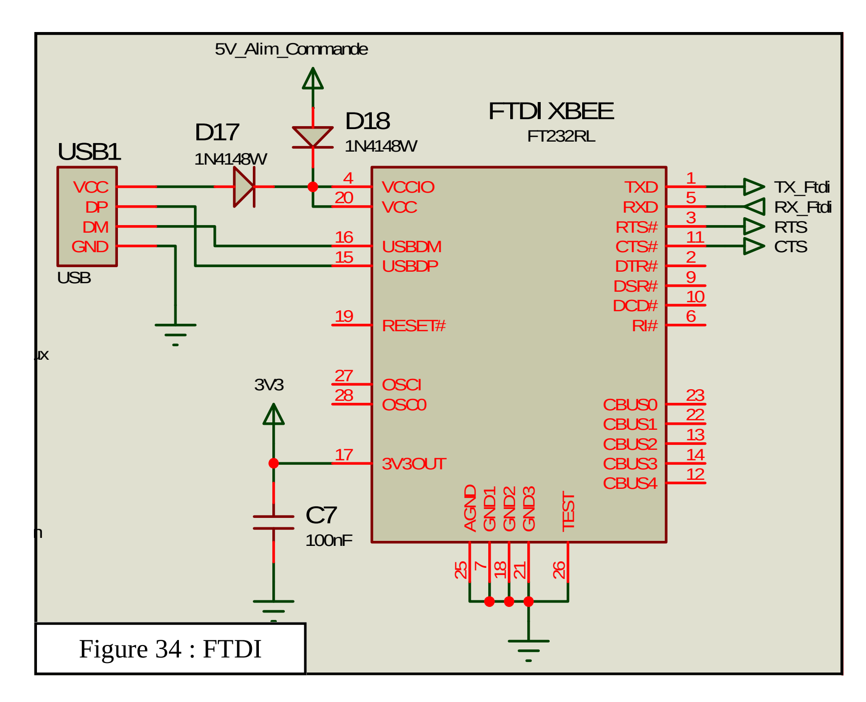
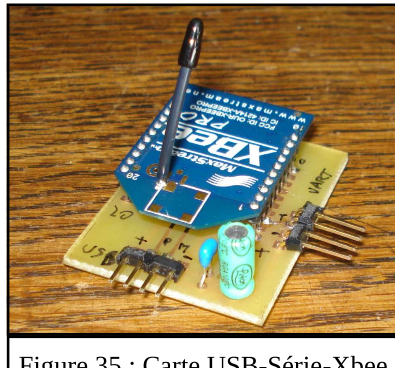


Figure 34 : FTDI


Figure 35 : Carte USB-Série-Xbee

Le FTDI est un composant très pratique. Grâce à ce composant, il a été possible de remplacer la grosse puce « USBMOD3 » précédemment installée sur la station sol V1. En regardant bien, l'USBMOD3 intègre lui-même un FTDI. Par ailleurs, le FTDI créé une alimentation de 3,3V que l'on peut réutiliser sur la carte pour alimenter le Xbee. On évite ainsi l'ajout d'un composant qui remplirait cette fonction. Cependant, cette alimentation 3,3V fluctue entre 3V et 3,6V avec un courant admissible de 50mA. Le Xbee PRO pouvant consommer un peu plus, il faudra donc faire attention à ce paramètre là. En effet, sur la datasheet, il n'est pas indiqué clairement les courants consommés. On peut cependant se référer aux valeurs des courants RX et TX à 3,3V (respectivement 55mA et 215mA). Lors de plusieurs essais, la carte de test "USB-Série -Xbee" (cf. Figure 35) a parfaitement bien fonctionné.

2) b - Câblage du Xbee :

Les niveaux fournis par le Xbee ne sont pas directement des signaux compatibles à ceux des PIC. Il fallait donc adapter la l'entrée IN du Xbee (5V→3.3V) ainsi que la sortie OUT du Xbee (3.3V→5V) :

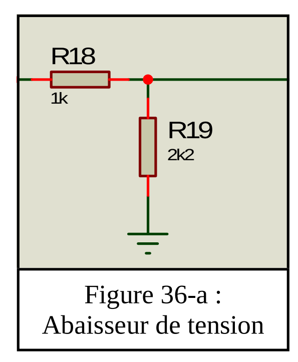


Figure 36-a : Abaisseur de tension

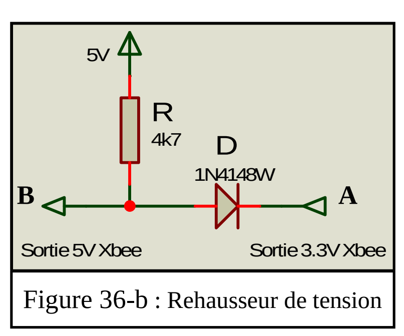


Figure 36-b : Rehausseur de tension

Pour la conversion 5V → 3.3V, l'adaptation s'est faite par un simple pont de résistance. (cf. Figure 36-a)

Et pour la conversion 3.3V → 5V , initialement, l'idée d'un montage rehausseur de tension (cf. Figure 36-b) utilisant une diode et une résistance de tirage avait été choisie et même malheureusement validée jusqu'au typon. Le principe de ce montage est le suivant : On choisit une diode dont la chute de tension vaut 1V. Cette diode sera toujours passante, ainsi, lorsque A = 3.3V, B vaut 4.3V, et lorsque A = 0V, B vaut 1V

En fait, il se trouve que ce montage risque de ne pas marcher aux vues des niveaux de tension des PIC et du Xbee. En regardant dans les datasheet du Xbee et du PIC, on peut retrouver les valeur limites des états hauts et bas. Voici ces valeurs (en Volts) : pour le Xbee : $V_{OH} = V_{cc} - 0,5$; $V_{OL} = 0,5$; Pour le PIC : $V_{IH} = 0,8.V_{DD}$; $V_{IL} = 0,2.V_{DD}$

Si un niveau haut émis par le Xbee est trop faible (2,8V), celui-ci sera rehaussé à 3,8V mais ne pourra pas être reconnu comme un état particulier par le PIC qui a une plage inconnue de 1V à 4V (lorsqu'il est alimenté à 5.0V). Par ailleurs, même l'état bas ne sera pas reconnu car il est trop haut.

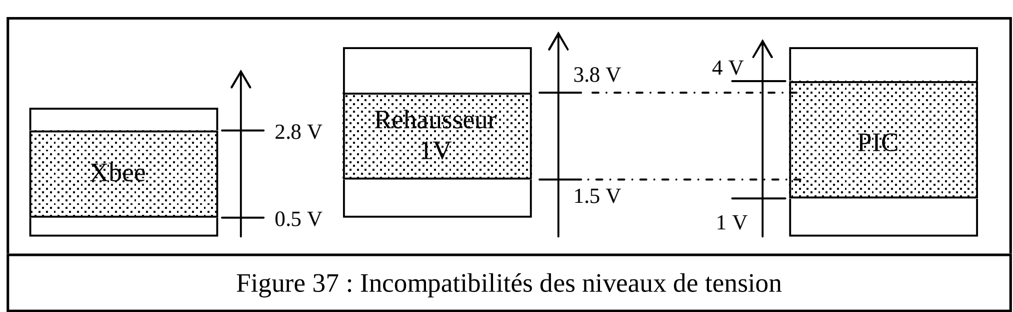


Figure 37 : Incompatibilités des niveaux de tension

À cause de ce désagrément, beaucoup de temps a été perdu. Pour réaliser cette adaptation, j'ai donc pensé au début utiliser un transistor mais il n'était pas possible d'utiliser un unique transistor bipolaire pour réaliser cette fonction (cf. Figure 38).

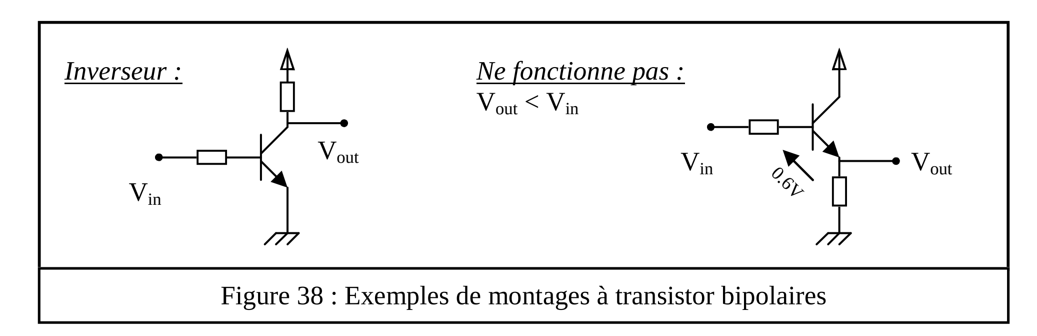


Figure 38 : Exemples de montages à transistor bipolaires

En ayant parlé à mes collègues de mon problème, Hugo Van Reeth (responsable carte capteur) m'a indiqué que lui utilisait une porte ET pour une adaptation quasi-identique. En effet une porte ET est un circuit qui admet un niveau limite pour une tension haute en entrée égale à $V_{IH} = 2.8V$. En sortie, la porte ET retourne un état haut à 5V compatible avec le niveau du PIC. Il s'agit donc là d'une excellente solution pour adapter un niveau 3.3V → 5V.

Une porte ET (Référence SN74AHCT1G08) a donc été soudée in extremis sur la face du dessous du circuit.

Cette partie de la carte a subi un grand nombre de modifications et est passée par plusieurs stades :

Au début, la porte ET était alimentée par le 5V général de la carte. (cf. Figure 39). Il y avait cependant des problèmes sur la communication Xbee ↔ USB qui persistaient. La démarche a donc été d'isoler petit à petit les signaux entrant/sortant du Ftdi. La configuration n'était possible que si la diode D15 était retirée (circuit ouvert entre Tx_6520 et Rx_Ftdi).

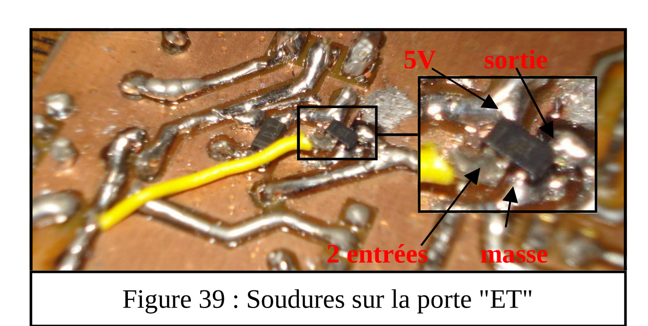


Figure 39 : Soudures sur la porte "ET"

C'est donc à ce moment là que j'ai compris que le problème venait de l'émission de données du PIC 6520. En effet, pour configurer le Xbee, il fallait alimenter la porte ET. Or, en alimentant cette porte ET, on alimentait également le reste de la carte et en particulier le PIC 6520.

Par la suite, il fallait donc trouver une façon d'alimenter la porte ET par une autre source de tension. Les deux diodes D17 et D18 (cf. Figure 34) servent justement à avoir une alimentation 5V pour le Ftdi (par USB) même lorsque le reste de la carte n'est pas alimenté. Il aurait donc fallu alimenter cette porte ET par cette alimentation qui se trouve sur le point commun des diodes D17 et D18 (Alimentation du Ftdi).

Voici une photo montrant le soudage "in extremis" de la porte ET ainsi que la modification d 'alimentation :

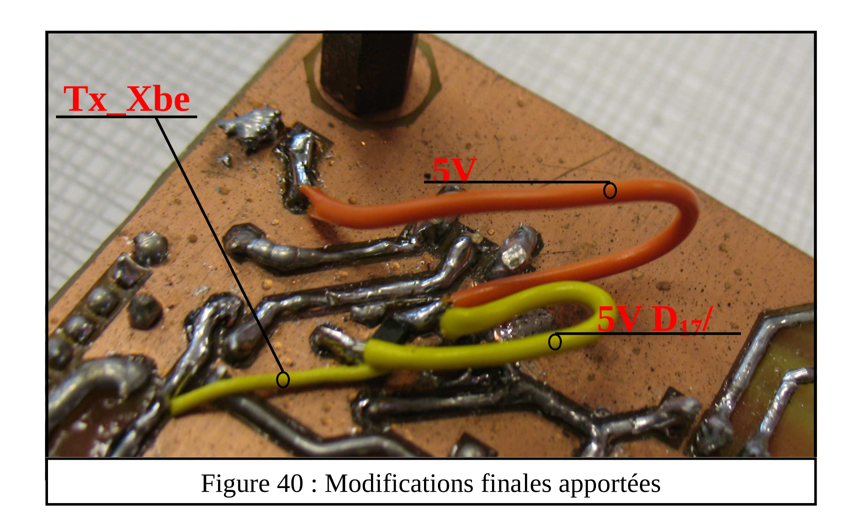


Figure 40 : Modifications finales apportées

Au final, le Xbee est câblé comme suit :

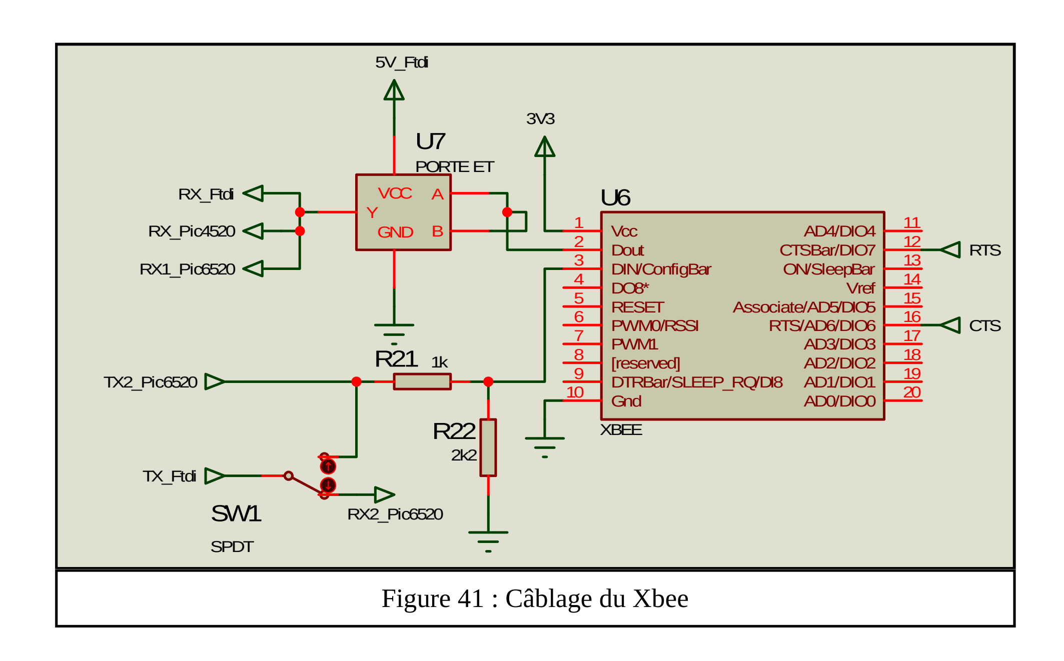


Figure 41 : Câblage du Xbee

La diode D14 sert à éviter que lors de la reprogrammation du Xbee, un signal puisse être envoyé sur le PIC 18F6520 (il ne s'agit que d'une sécurité). R18 et R19 forment le pont diviseur de tension 5V → 3.3V. La porte ET se charge donc de rehausser la tension 3.3V → 5V. Et enfin, pour permettre de le reprogrammer sans avoir à débrancher le Xbee, un interrupteur a été ajouté. Cet interrupteur déconnecte le 18F6520 de l'USB PC, pour permettre la communication bidirectionnelle entre le PC (USB) et le Xbee (sans cet interrupteur, on n'aurait que le sens Xbee→PC).

2) c - Câblage des PICs

Les PICs sont cadencés à 12 MHz par un quartz extérieur. En plus de leurs connectiques essentielles (capteurs, commande des puces, etc…), ils sont tous les deux équipés d'un jeu de 6

LEDs de débogage (4 vertes et 2 rouges). Le connecteur faisant office de programmateur PicKit2 peut également servir de débogage par UART : seul un interrupteur doit être basculé (Figure 46). Les deux PICs sont découplés par un condensateur CMS de 100nF placés sur le typon au plus proche des PICs.

Les ponts diviseurs formés des résistances R13 à R16 permettent d'adapter le signal 12V issu des capteurs de inductifs montés sur l'antenne (cf. Figure 42). La résistance R12 est bien entendu une résistance de tirage évitant un court circuit de l'alimentation lors de l'appui sur le bouton. Ce bouton est le bouton "Init" d'initialisation de la station sol.

Lors du développement de la carte, une idée a été émise concernant une initialisation complètement automatisée de la station grâce à un compas (une boussole). Cette boussole (cf. Figure 43) a été développée par Emmanuel Roussel. Pour plus de détails concernant une implémentation de celle-ci, il faudrait voir le rapport la concernant.

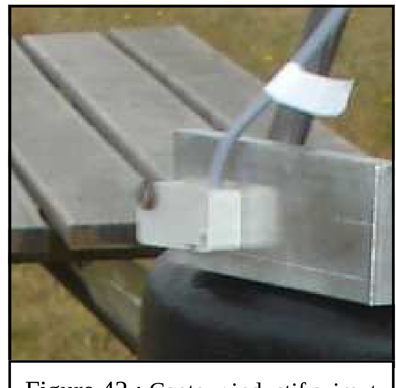
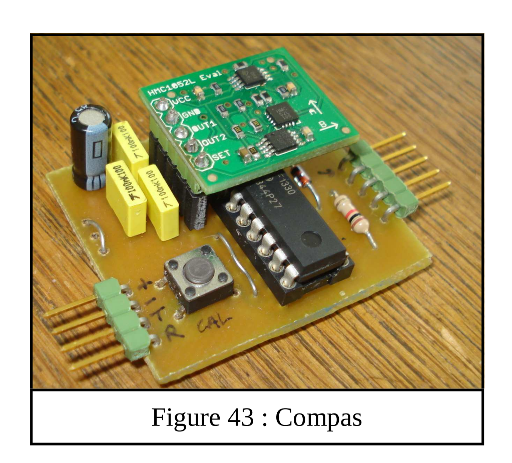



Figure 42 : Capteur inductif azimut


Figure 43 : Compas

Voici le schéma de câblage du PIC de commande des moteurs :

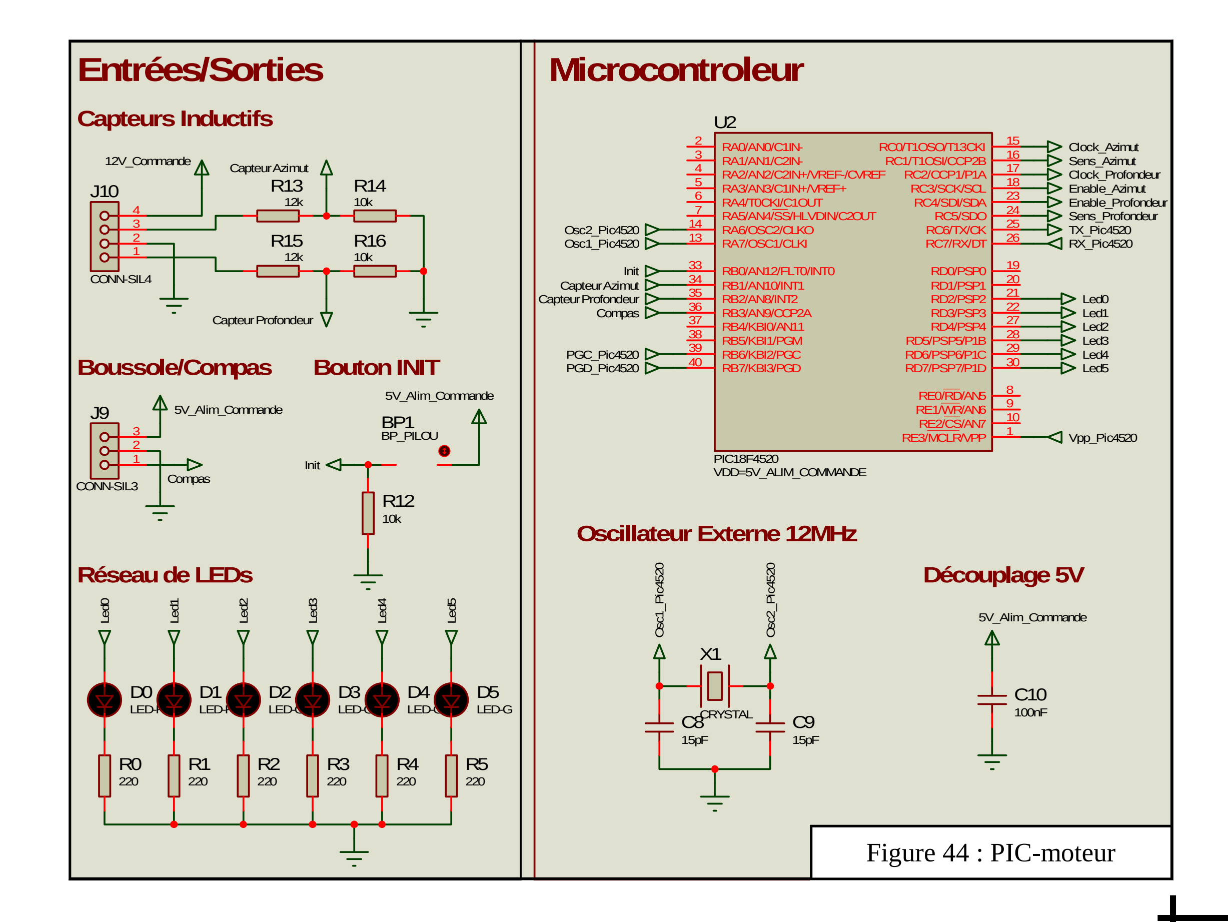


Figure 44 : PIC-moteur

Concernant le PIC d'aiguillage, le joystick est branché directement sur les pins CCP (Capture Compare). Ces pins permettent d'analyser facilement des signaux PWM.

Voici le schéma de câblage du PIC d'aiguillage :

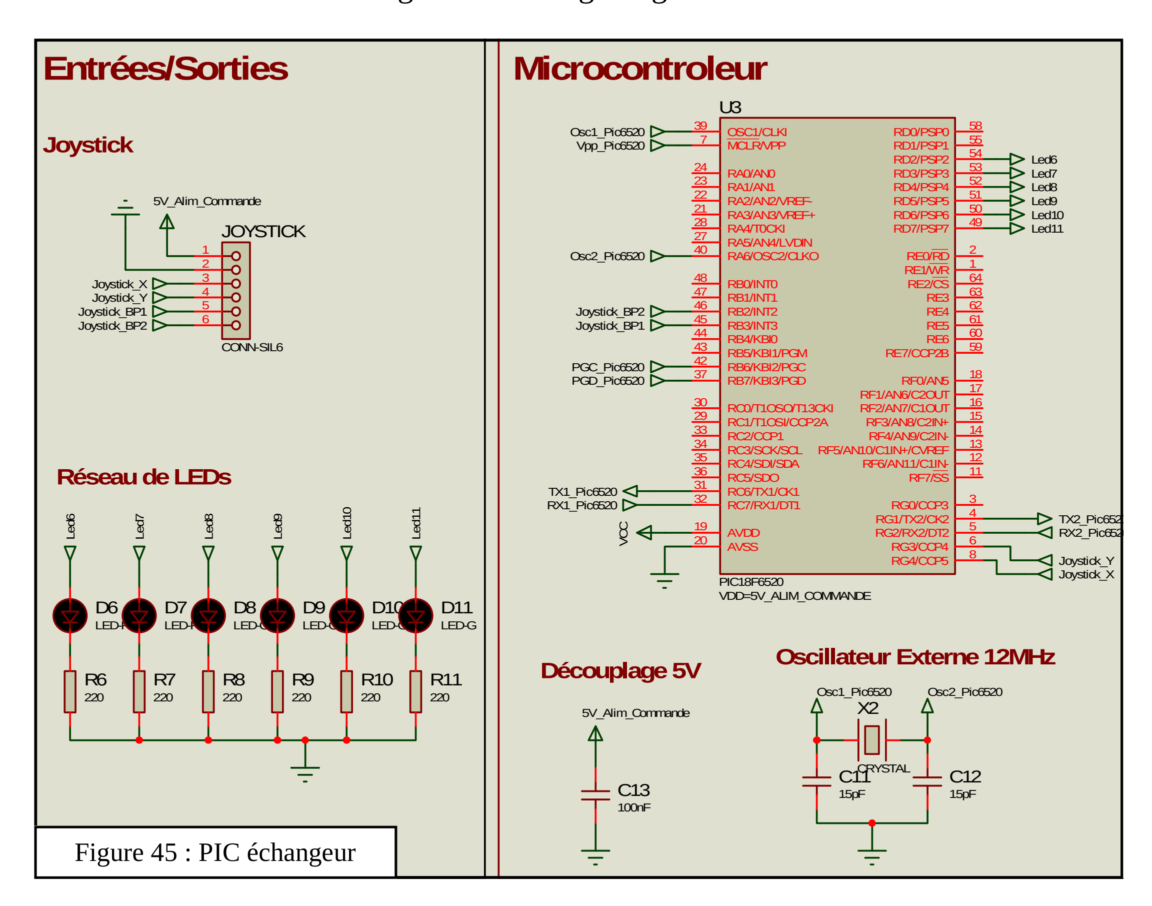


Figure 45 : PIC échangeur

Voici le schéma des connecteurs de programmation intégrant un port UART :

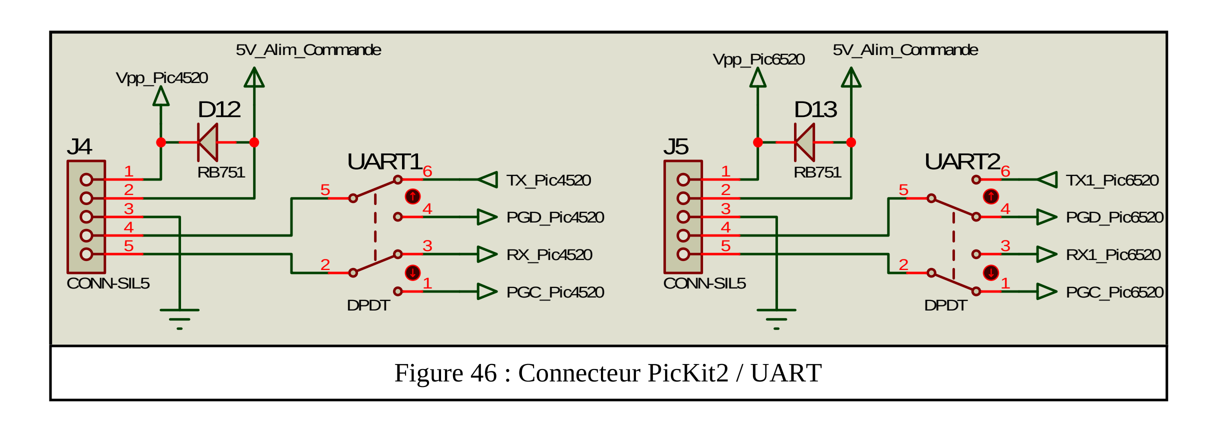


Figure 46 : Connecteur PicKit2 / UART

Les diodes D12 et D13 sont souvent utilisées sur ces connecteurs pour imposer un état haut sur la patte du $\overline{MCLR}$ tout en évitant un problème de niveau de tension (entre le 5V régulé et le 5V issu de l'USB PC) lors de la programmation. Il n'y a pas de critères spéciaux pour le choix de ces diodes : il faut juste avoir une diode CMS pas chère de tension de seuil inférieure à 1V (pour que la pin $\overline{MCLR}$ puisse lire un état haut : 4V est le minimum possible).

2) d - Connecteur RJ11 :

Sur la carte commande de la station sol V1, le connecteur avait mal été câblé puisque pour faire le lien avec le module de surimpression vidéo, il fallait croiser le câble RJ11 ce qui n'est pas très pratique. Pour cette raison, la nouvelle carte intègre un connecteur RJ11 qui lui est cette fois adapté au module de surimpression. Un câble droit devra donc être fait.

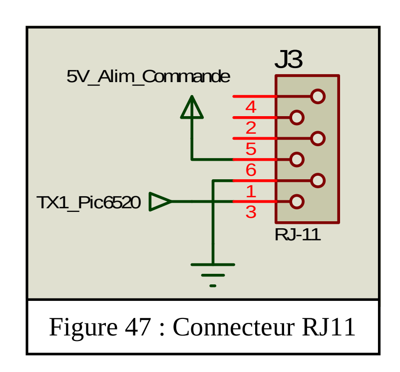


Figure 47 : Connecteur RJ11

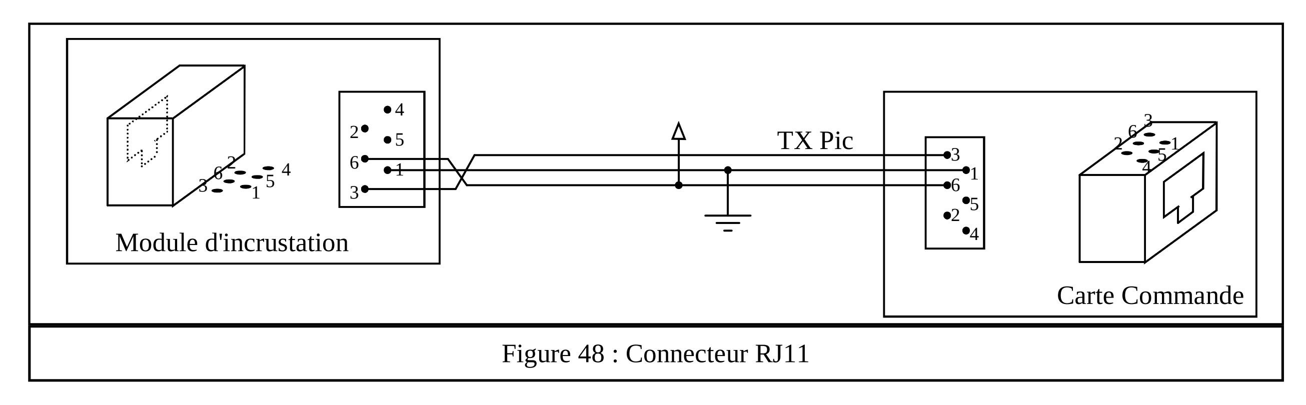


Figure 48 : Connecteur RJ11

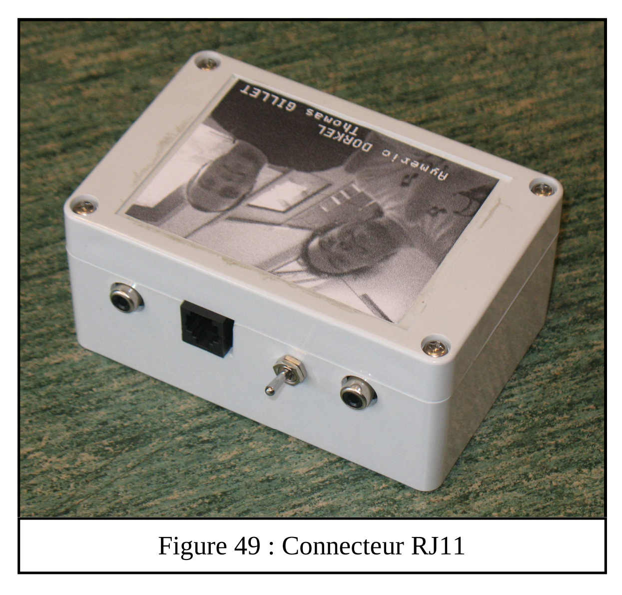


Figure 49 : Connecteur RJ11

2) e - Connecteur DIL10 Carte Commande/Puissance :

Contrairement à la station sol V1, le connecteur intègre ici une alimentation 12V issue de la carte puissance. Précédemment, cette alimentation était fournie par un deuxième câble d'alimentation directement depuis la batterie. L'alimentation 5V a aussi été intégrée à ce connecteur, ce qui permet de n'utiliser plus qu'un seul régulateur 5V qui se trouve sur la carte puissance. En revanche, les signaux "*Reset*" des deux puces L297 ont été retirés car ils ne sont pas utiles dans notre cas (cf. la suite de cette note d'application). Il s'agit en effet de signaux permettant de réinitialiser l'état initial des bobines à commander. Toute la commande se faisant de façon transparente pour nous, ces signaux ne nous sont pas vraiment nécessaires.

Un interrupteur permet d'alimenter le 5V de la carte commande soit par le 5V régulé de la carte puissance, soit par n'importe quelle broche 5V de la carte commande (exemple PicKit etc…) . Une LED permet de visualiser si la carte puissance est alimentée ou non. Sur les schémas

de "correction", une deuxième LED a été rajoutée pour indiquant l'alimentation de la carte commande. L'alimentation 12V est découplée par un condensateur CMS de 100nF en sortie de connecteur, et pour le 5V, il est régulé en sortie d'interrupteur.

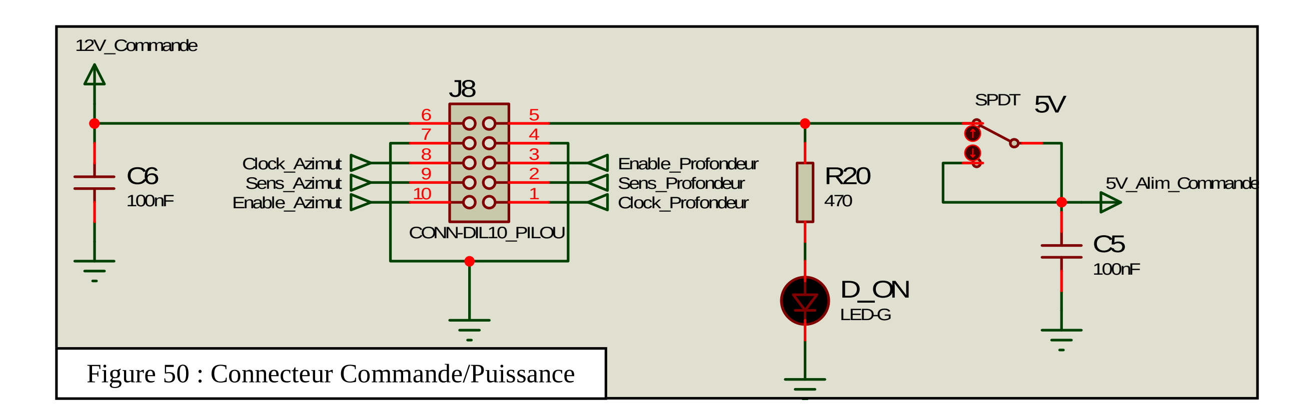


Figure 50 : Connecteur Commande/Puissance

2) f - Gestion de l'alimentation :

On utilise directement une tension 12V issue d'une batterie qui une fois filtrée alimente directement les hacheurs des moteurs. Une diode a été ajoutée en série pour éviter une inversion dans le sens des connecteurs. Pour former une tension 5V, un simple régulateur 5V a été choisi. Il permet d'alimenter les PICs de la carte commande, ainsi que les puces L297 et L298. Plus tard, la question du dimensionnement du régulateur a été abordée. Le régulateur 5V alimente au maximum 3 pics (les 2 principaux ainsi que celui du Joystick), 2 composants L297 et 2 L298.

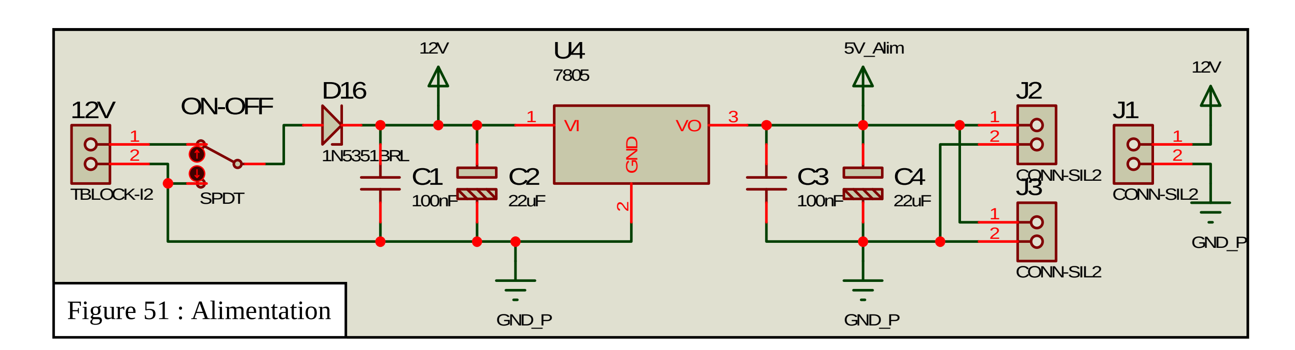


Figure 51 : Alimentation

Concernant les PICs, ils sont limités à 250mA chacun (cf. page 321 et 307 du 18F4520 et 18F6520), les L298 consomment au repos 36mA maximum chacun (cf. page 3 de la datasheet), et enfin, pour les L297, aucune indication n'est donnée. La détermination précise de la valeur maximale du courant que le régulateur devra fournir est difficile. Cependant, cette valeur est déjà majorée si on prend effectivement 250mA pour les PICs. En effet, ceux-ci ne consommeront pas autant puisque tous leurs périphériques ne sont pas utilisés. Par conséquent, on peut retenir pour valeur haute une consommation de 900mA.

Or les simples régulateurs 7805 permettent de délivrer un courant de 1A (voire plus avec un dissipateur adapté). De toutes façons, vu le différentiel de tension 12V → 5V, un dissipateur devrait être ajouté à ce composant. Pour informations, si jamais le choix de ce régulateur vient à être remis en question, le remplacement par un LM323T (dans le même package TO-220) pourra être envisagé.

Un dissipateur pour ce composant devra dissiper :

$$(U_{batterie} - U_{régulée}) * I_{consommée} = (12\text{-}5)*1 = 7 \text{ Watt}.$$

Si l'on considère une température ambiante de 25°C, et que l'on souhaite obtenir une température de jonction de 30°C, on a : $\Delta T = P*R_{dissipateur}$ soit $R_{dissipateur} = 5°C/7W$

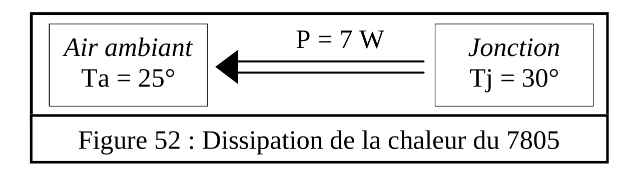


Figure 52 : Dissipation de la chaleur du 7805

Il faudra donc un dissipateur de résistance thermique au maximum de 0,7 °C/W (une valeur basse est préférable). Lors de hautes températures (ex à 40°C), la jonction sera à environ 35°C.

Concernant l'alimentation, trois connecteurs supplémentaires (2x5V et 1x12V) ont été ajoutés sur la carte puissance dans le cas où un accès à une de ces tensions serait nécessaire. Sur la carte commande, il y a également des points test (masse;tension) permettant d'accéder aux tensions 12V, 5V et 3.3V.

2) g - Commande des moteurs :

Nous avons 2 moteurs pas à pas à contrôler. Un moteur pas à pas est un ensemble de bobinages que l'on peut alimenter indépendamment les uns des autres. En alimentant certaines bobines du stator, on peut donc orienter le rotor (aimant permanent) selon des directions privilégiées. Selon le nombre de phases du moteur et le nombre de paires de pôles, un moteur pas à pas possède un certain nombre de positions privilégiées. En alimentant donc certaines bobines les unes après les autres, on peut donc faire avancer le moteur de pas en pas (cf. Figure 53).

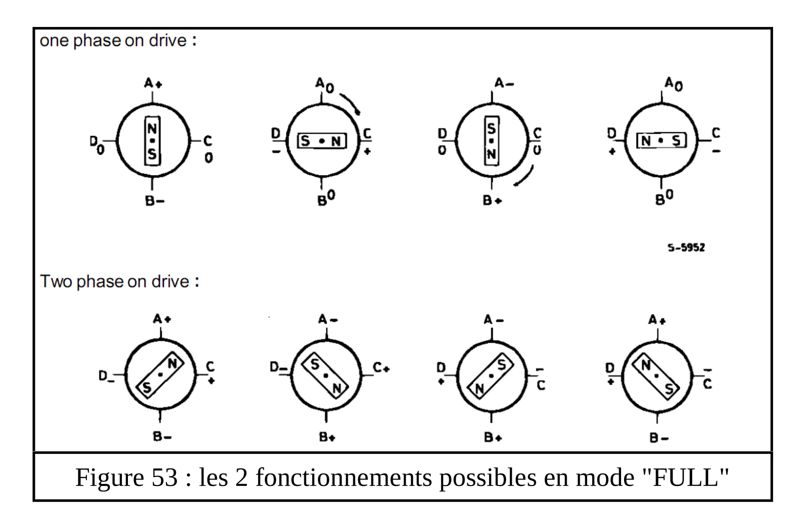


Figure 53 : les 2 fonctionnements possibles en mode "FULL"

Pour alimenter un moteur on utilise très souvent des hacheurs intégrés de type L298. Ici, il s'agit d'un moteur pas à pas. La commande n'est pas forcément évidente. Il existe des composants L297 qui remplissent la fonction de driver de moteur pas à pas.

Le schéma en Figure 27 est issu de la datasheet du L297 qui propose un schéma tout fait de commande de moteur pas à pas. À ce schéma de base, certains ajouts ont été faits : un potentiomètre de contrôle du courant admissible a été ajouté (cf. Figure 55) ainsi qu'un interrupteur permettant de sélectionner le mode de commande du hacheur L298 : $\overline{INH1}$, $\overline{INH2}$ ou bien A,B,C,D. Ces deux modes de commande sont expliqués dans la datasheet.

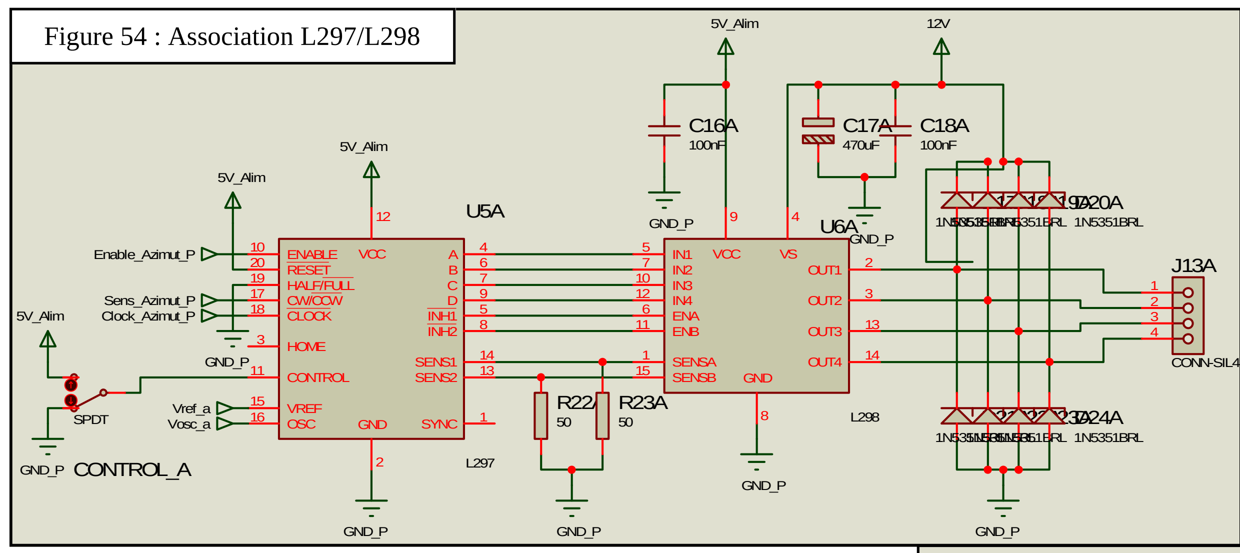


Figure 54 : Association L297/L298

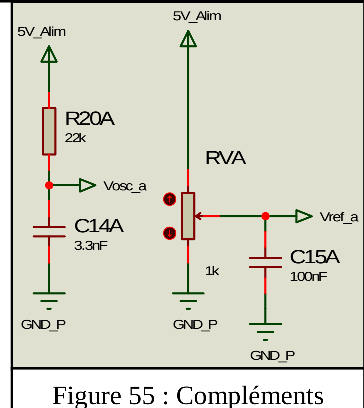


Figure 55 : Compléments

Au final, il se trouve qu'il n'y a pas de grande différence entre ces deux modes. On pourrait donc à l'avenir penser à tirer la patte 11 du L297 à 5V.

Le filtre RC est un mécanisme de protection qui contrôle le courant des moteurs en faisant varier le rapport cyclique du hacheur. Ce n'est pas la même chose que l'entrée Clock qui elle contrôle la vitesse du moteur : à chaque front montant du signal Clock, l'alimentation des phases du moteur change (cf. Figure 53).

La pate 19 est à la masse, et donc la puce va fonctionner en mode "FULL", c'est-à-dire qu'il y aura 4 états successives des sorties A,B,C,D. Selon l'état initial du "translator" (composant principal du L297), on aura soit le fonctionnement "one-phase-on" soit le fonctionnement "two-phase-on" (cf. Figure 53).

Les résistances R22 et R23 sont des résistances "SENSE" de mesure du courant entrant dans les moteurs. Les puces L297 sont équipées de comparateurs de tension entre la tension aux bornes de ces résistances U_{sense} et la tension V_{ref} appliquée sur la patte 15 du circuit. Si la tension U_{sense} dépasse V_{ref} , alors, la puce modifie le rapport cyclique pendant un certain temps de façon à faire chuter la tension et donc le courant tiré par les moteurs.

Les résistances "SENSE" se fixent de manière à avoir une tension comprise entre 0 et 5V. Avec la formule $U_{sense} = R_s*I_{adm}$, et avec $R_s = 1\Omega$, on aura donc une tension de $U_{sense} = 2V$ lorsque le moteur consomme 2A. Pour pouvoir contrôler le courant admissible on place le potentiomètre RV (cf. Figure 55) sur l'entrée Vref. Pour limiter à 2A la consommation, il faudra donc tourner le potentiomètre jusqu'à obtenir une tension de 2V sur la patte 15 du L297. Il faudra bien veiller au calibrage de ces potentiomètre car si ils sont réglés à une tension maximale pour V_{ref}, alors, les moteurs pourront tirer 10A, ce qui risque d'être assez dangereux.

3) Réalisation des cartes :

Afin de gagner en place, la station sol sera réalisée comme précédemment sur deux cartes "Commande" et "Puissance". Deux circuits intégrés ont été choisis en CMS car il n'y avait pas le choix (le Ftdi ainsi que le PIC 18F6520). D'autres composants plus classiques (résistances, diodes, condensateurs, LEDs) ont également été choisis en CMS.

Le PIC "moteur" 18F4520 aurait pu être choisi en CMS. Cependant, la carte étant nouvelle (prototype et non une version finale), et le tirage de celle-ci n'étant pas professionnel, le format DIL standard a été préféré à celui "CMS". Sur les cartes présentées ci après, les plans de masse ont été retirés.

3) a - Typon carte commande :

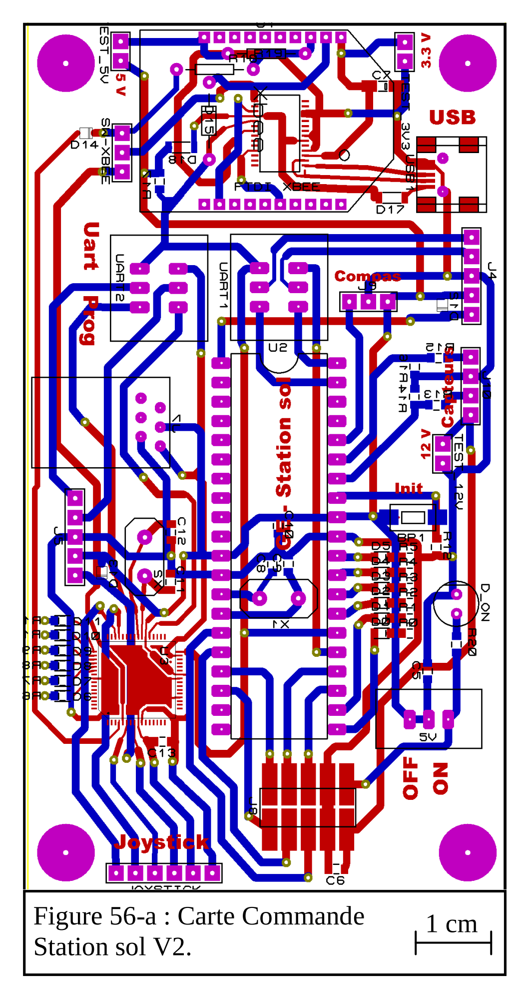


Figure 56-a : Carte Commande Station sol V2.

1 cm

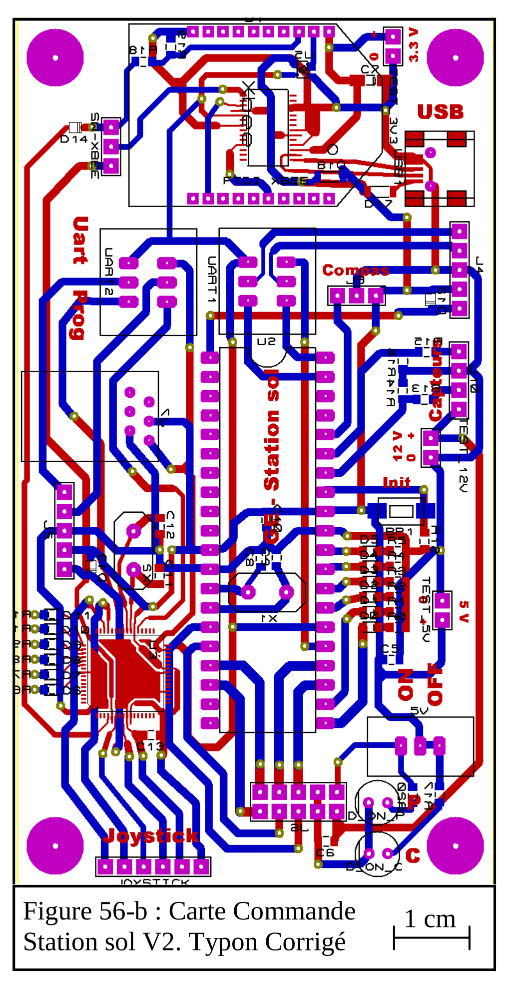


Figure 56-b : Carte Commande Station sol V2. Typon Corrigé

1 cm

Après débogage de la carte, le typon a été modifié. En effet, voici un listing des défaut relevés sur ce premier typon :

- Problème avec le montage rehausseur de tension (Diode+Résistance), remplacé par une porte ET à alimenter par le point commun des diodes D17 et D18 (ce qui n'est pas fait sur la V2 modifiée).
- Le package du connecteur DIL10 n'est pas bon si on utilise un plan de masse (toutes les pattes étaient reliées). Remplacement de ce package par le package d'origine en utilisant l'option "Change Layer → Solder Side"
- Le connecteur RJ11est mal situé car lorsque l'on superpose cette carte avec la carte puissance, ce connecteur est mal situé (au niveau du radiateur du hacheur du moteur de profondeur).
- Une deuxième LED d'état d'alimentation de la carte commande a été ajoutée.

3) b - Typon carte puissance :

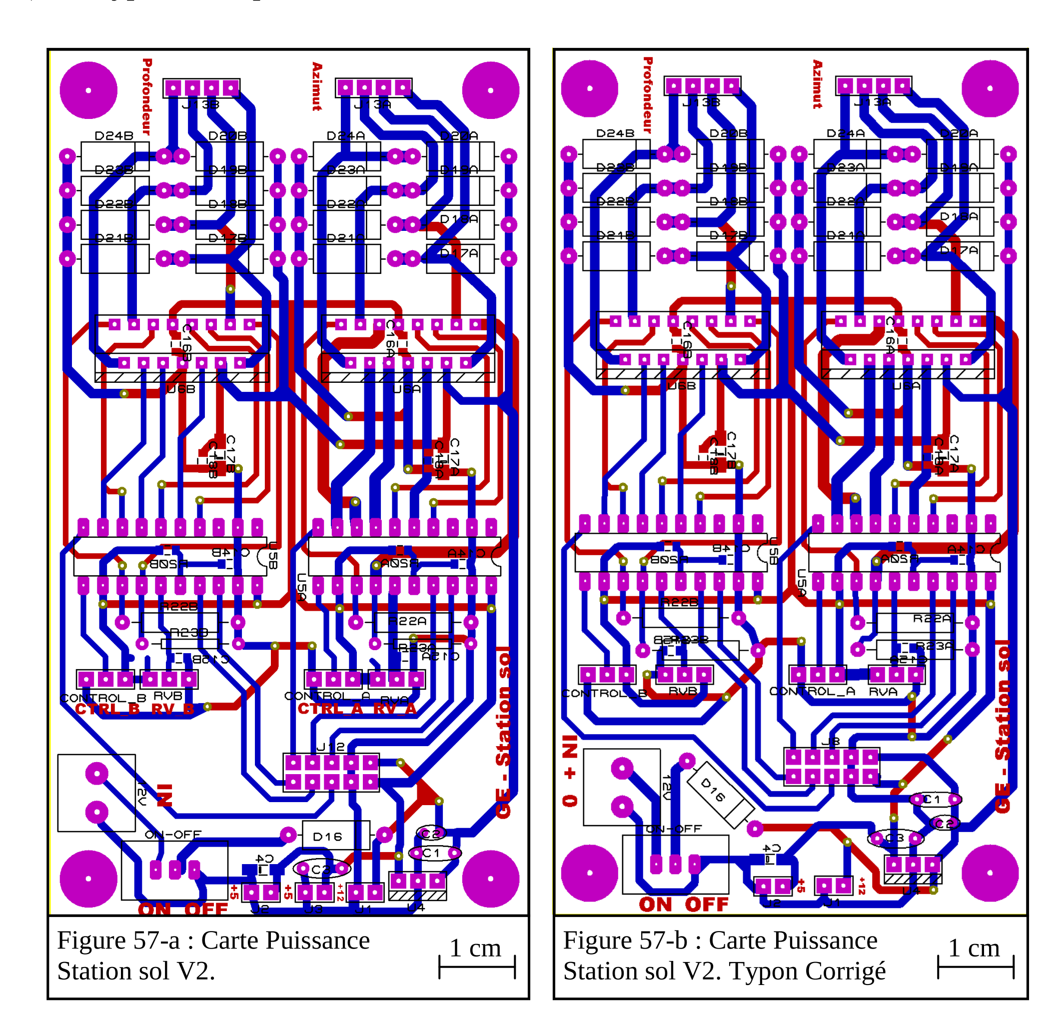


Figure 57-a : Carte Puissance Station sol V2. 1 cm

Figure 57-b : Carte Puissance Station sol V2. Typon Corrigé 1 cm

Cette carte possède elle aussi quelques défauts :

- Package des Résistances "SENSE" R22 et R23.

- Le régulateur est trop proche des connecteurs autour. Il faudra prévoir un emplacement pour un dissipateur.

- Vérifier le package des condensateurs CMS

- Il faut éloigner les potentiomètres à la fois du switch "CONTROL" ainsi que des résistances 22 et 23. Il faudrait également tourner de 180° un de ces potentiomètres afin que le réglage du courant admissible se fasse dans le même sens pour les deux étages moteurs. (Par exemple si on visse, on augmente le courant limite, si on dévisse, on diminue cette limite). Pour plus de clarté, on pourrait faire apparaitre dur le package la vis de réglage.

- Le changement du bornier d'alimentation générale 12V pourrait être envisagé. On pourra choisir une solution à clips, plus pratique pour le montage/démontage.

- Pourquoi pas supprimer les interrupteurs "CTRL_x" car comme dit en page 13, il ne sont pas utiles.

3) c - Prototype de la station :

Voici une image présentant les emplacements des différents composants/connecteurs. On remarquera qu'un soin particulier a été apporté à la logique de placement de ceux-ci.

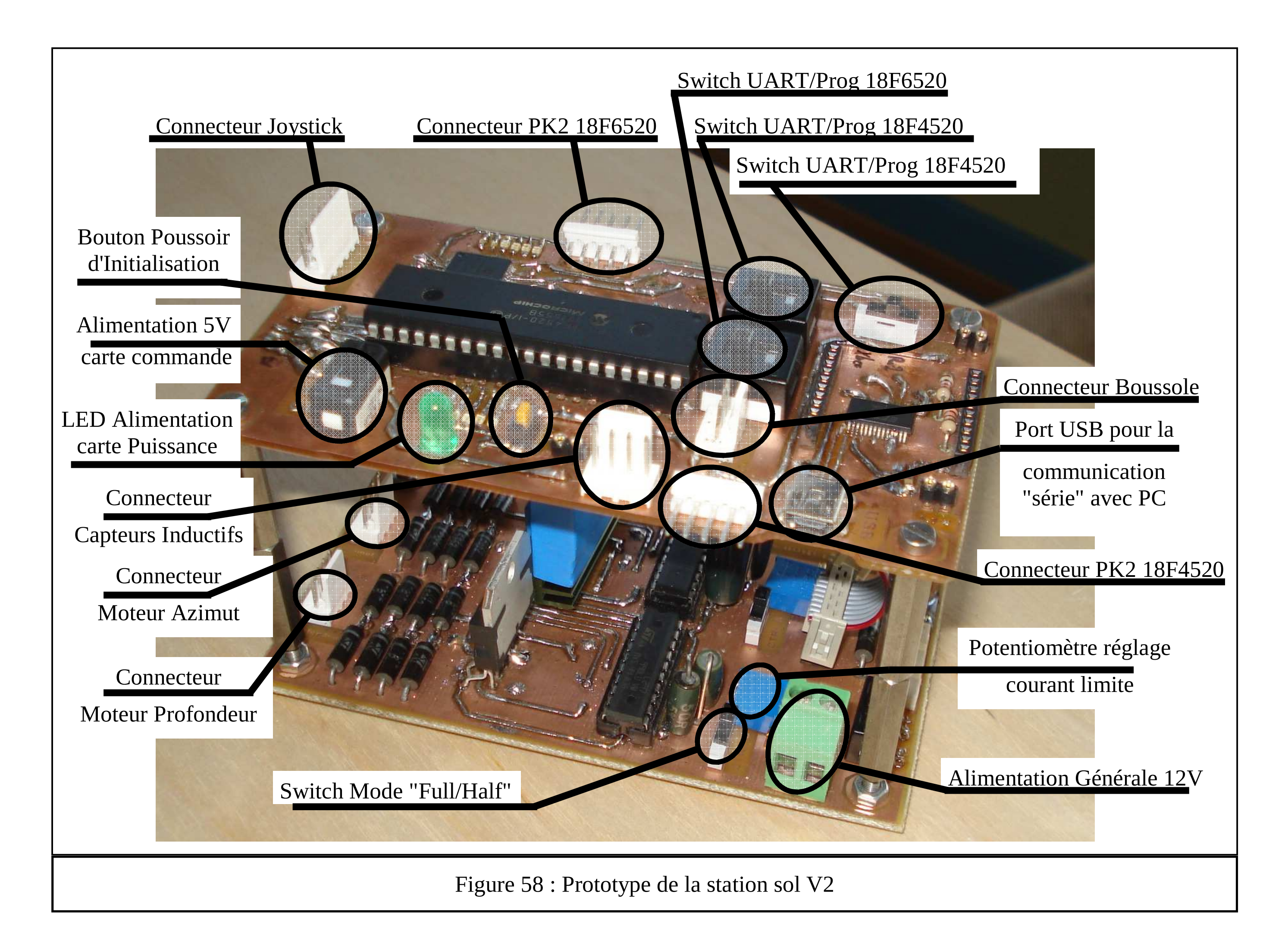


Figure 58 : Prototype de la station sol V2

3) d - Commande des composants :

La commande a été effectuée sur le site de Farnell. Voici la liste de composants nécessaires pour ces circuits :

1 x PIC 18F4520
1 x PIC 18F6520
1 x Ftdi FT232RL
1 x Xbee Pro
1 x 7805
2 x L297
2 x L298
1 x Porte ET Cms
2 x Quartz HS 12MHz

4 x Condensateurs Cms 15pF
12 x Condensateurs Cms 100nF
1 x Condensateur Plastique 100nF
2 x Condensateur Chimique 22µF
2 x 470 uF
3.3nF

8 x LEDs Cms Vertes
4 x LEDs Cms Rouges
2 x LEDs Simple Vertes

12 x Résistances Cms 220Ω
3 x Résistances Cms 10kΩ
2 x Résistances Cms 12kΩ
1 x Résistance Cms 1kΩ
1 x Résistance Cms 2.2kΩ
2 x Résistance Cms 22kΩ
4 x Résistance de Puissance 0.5Ω
2 x Résistances LEDS xxxxxxxxxx
2 x Potentiomètre 10kΩ

5 x Diodes (chute de tension faible)
17 x Diodes (Roue Libre et Alim)

2 x Connecteurs SIL5 (pas 2.54mm)
1 x Connecteur SIL3 (pas 2.54mm)
3 x Connecteur SIL4 (pas 2.54mm)
1 x Connecteur SIL6 (pas 2.54mm)
2 x Connecteurs SIL2 (pas 2.54mm)
2 x Connecteurs DIL10 (pas 2.54mm)
2 x Connecteurs Femelle DIL10 (pas 2.54mm)
1 x Connecteur RJ11
1 x Bornier d'Alimentation Générale
3 x Barrettes sécables SIL2 (pas 2.54mm)
2 x Barrettes sécables SIL10 (pas 2.00mm)
1 x Port USB
2 x Petits Interrupteurs SPDT
2 x Interrupteurs SPDT
2 x Interrupteurs DPDT
1 x BP Cms

3) e - Tests effectués sur la carte puissance :

Afin d'effectuer une première vérification du fonctionnement de la carte puissance, un signal Clock issu d'un GBF a été imposé sur le connecteur DIL10 et une mesure des signaux de sortie ont été effectués. Lors de ces tests, les moteurs ont bien fonctionné comme il le fallait. Voici sur les figures 19 et 20, les tensions relevées sur l'entrée Clock de la puce L297 ainsi que sur les bobines A, B, C, D du moteur

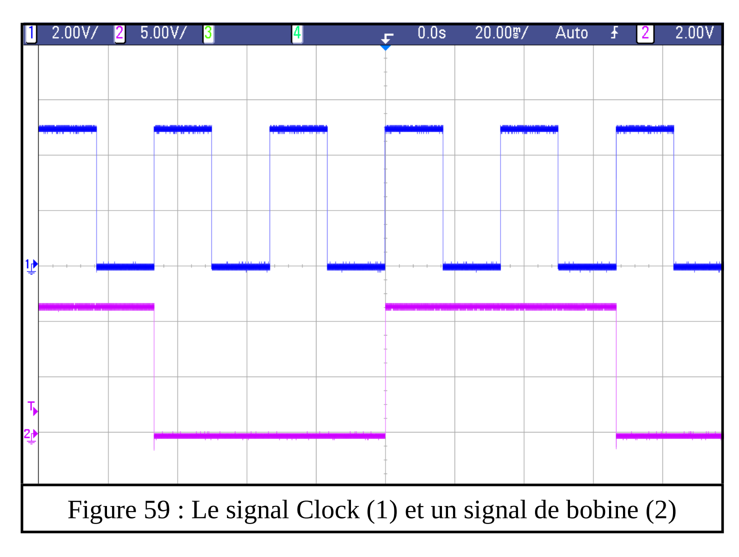


Figure 59 : Le signal Clock (1) et un signal de bobine (2)

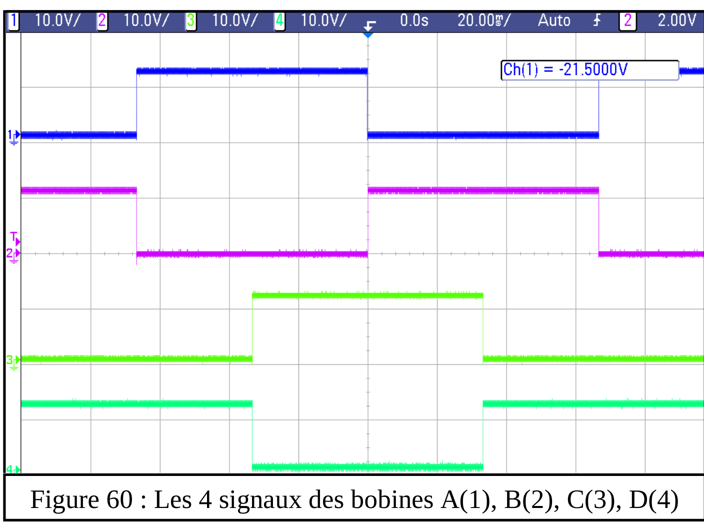


Figure 60 : Les 4 signaux des bobines A(1), B(2), C(3), D(4)

V - Descriptif du programme :

Il y a deux programmes sur cette station sol car deux microcontrôleurs. Le microcontrôleur responsable de la commande des moteurs est le 18F4520, tandis que celui qui fait l'aiguillage des données est le 18F6520. Pour le moment, seul le programme des moteurs a été fait.

1) PIC motorisation :

Le but de ce code est de prendre en compte la position du drone (coordonnées GPS et altitude) et après une phase d'initialisation (coordonnées de référence de la station sol), les moteurs pas à pas sont commandés en nombre de pas à effectuer pour s'incliner en direction du drone sachant qu'un tour d'azimut vaut 488 pas et un tour profondeur vaut 246 pas.

1) a - Organigramme :

Voici un organigramme très simplifié du code :

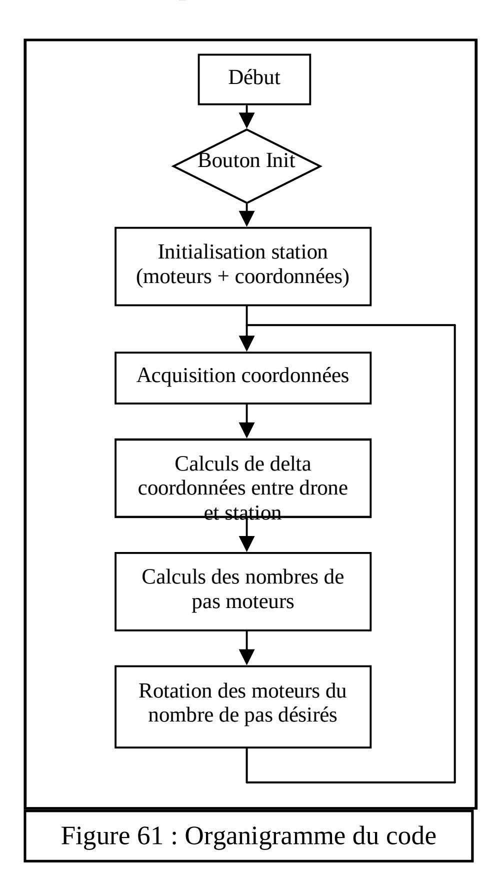


Figure 61 : Organigramme du code

1) b - Fonctionnement du code :

Actuellement, la réception de la trame se fait sous interruptions : dès que l'on active le bit d'activation des interruptions réception UART (*RCSTAbits.CREN*), les valeurs de longitude, latitude et d'altitude du drone sont chargées dans les variables correspondantes (*longitude_GPS*, *latitude_GPS* et *altitude_GPS*).

Une fois ces valeurs chargées (ce qui se produit lorsque la fonction *acquisition_GPS()* est appelée), la position des deux moteurs étant connue (variables *position_moteur_azimut* et

position_moteur_profondeur), le programme va calculer le nombre de pas à effectuer pour chacun des deux moteurs.

En premier lieu, le moteur azimut est commandé (*tracking_azimut(1)*) puis c'est au tour du moteur de profondeur (*tracking_profondeur(5)*). Dans le code des fonctions de "tracking", une horloge est créée. Cette horloge est nécessaire, et c'est elle qui va faire bouger le moteur d'un pas à l'autre à chaque front montant. À chaque cycle de cette horloge, les variables de position des moteurs sont incrémentées (ou décrémentées selon le sens de rotation).

Les étapes de suivi des deux moteurs sont actuellement séquentielles : on charge la trame, on calcule le nombre de pas on bouge les moteurs, puis on recommence. Le système fonctionne donc en saccadé. Il serait beaucoup plus intéressant de faire une lecture en continu de la trame, ainsi les moteurs tourneront en continu, ce qui permettra de rendre le système plus fluide. Cela dit, les moteurs sont assez rapides pour arriver à la position souhaitée très rapidement, et à l'échelle du déplacement du drone, les moteurs devraient fonctionner assez bien dans l'état actuel des choses. C'est d'ailleurs le cas actuel : lors du suivi, il n'y a pas de décrochage dû à une vitesse trop grande du drone.

<u>1) c - Stratégie de calcul des pas moteurs :</u>

Le calcul de l'angle du drone est donné par la simple formule suivante (cf. la figure 35)

$$\text{angle} = \arctan\left(\frac{\text{longitude_GPS - longitude_station}}{\text{latitude_GPS - latitude_station}}\right)$$

Une fois la valeur calculée, il faut également savoir dans quel cadran le drone se trouve. En effet, la fonction arctan() est une fonction de] -∞ ; +∞ [vers] -π/2 ; π/2 [.

Sur la Figure 63, la valeur des angles calculée est donnée par *valeur* et la valeur de l'angle réelle est donnée par *angle*. Au final, la fonction utilisée retourne un angle compris dans l'intervalle] -π ; π [(sachant que l'azimut Nord correspond à l'angle 0).

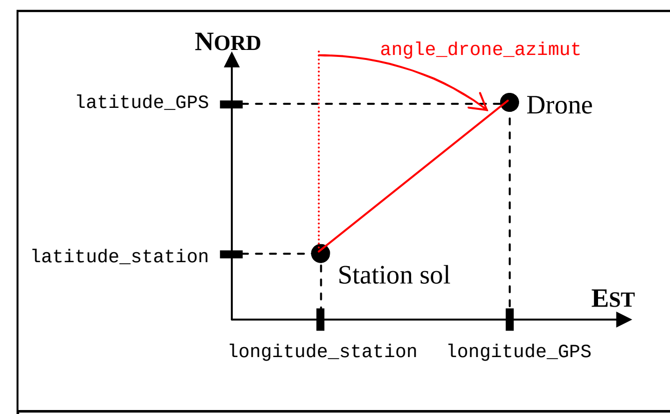


Figure 62 : Schéma pour le calcul de l'azimut

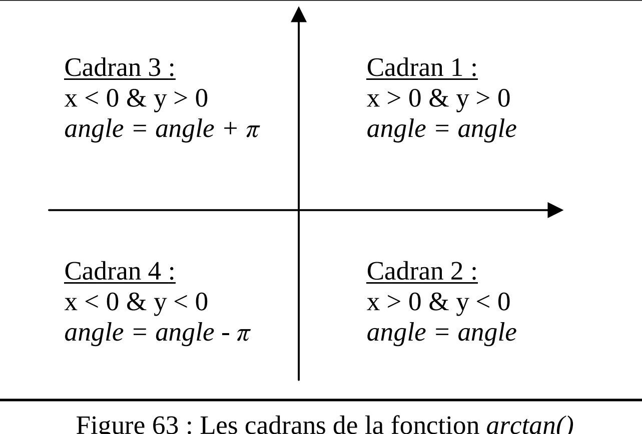


Figure 63 : Les cadrans de la fonction *arctan()*

Une fois cet angle obtenu, il reste à calculer le delta angle entre cet angle et celui où se trouve le moteur. L'angle du moteur est donnée par la formule :

$$\text{angle_moteur_azimut} = \text{position_moteur_azimut} . \frac{2.\text{pi}}{488}$$

Concernant la soustraction, il faut s'assurer que la différence est belle et bien inférieure à π puis savoir de quel signe est cette différence. Le signe sera donné par une variable spécifique (*sens_rotation_azimut*)

Pour ce qui est du calcul de l'angle de la profondeur, il faut prendre en compte la distance de la projection au sol de la station sol et du drone (cf. la cotation "distance" sur la Figure 64). L'angle de profondeur sera finalement donné par la formule :

$$angle = \arctan\left(\frac{altitude_GPS - altitude_station}{distance}\right)$$

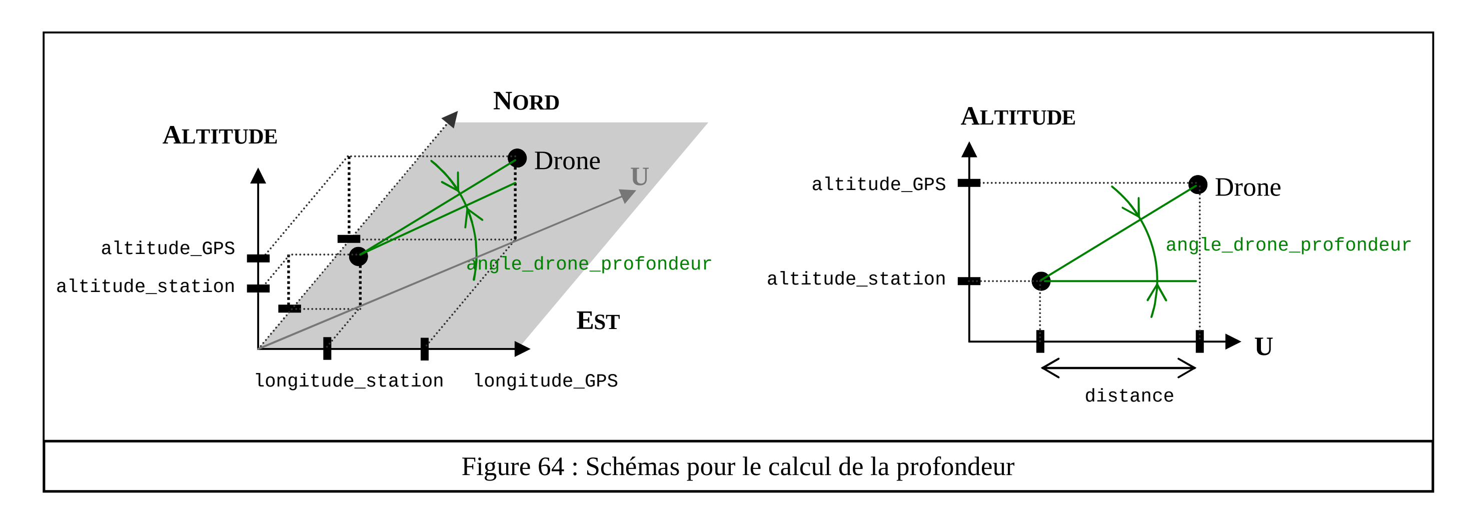


Figure 64 : Schémas pour le calcul de la profondeur

1) d - Détails du code :

Afin de permettre une plus grande rapidité de prise en main du code de la station à l'avenir, voici des extraits détaillés du code (certains passages ont été supprimés/modifiés pour plus de lisibilité) :

La fonction main :

```
void main(){

   /* déclaration de variables  */      /* ...... */

   /* Initialisation du PIC (UArt, Interruptions, TRIS, etc */    /* ...... */

   while(INIT == 0 ) ;
   initialisation_station() ;
   while(1) {
       acquisition_GPS() ;
       if ((latitude_GPS!=0)&&(longitude_GPS!=0)) break ;
   }
   longitude_station = longitude_GPS ;
   latitude_station = latitude_GPS ;
   altitude_station = altitude_GPS ;

   while(1) {

       acquisition_GPS();
       coord_x = (long) longitude_GPS - (long) longitude_station ;
       coord_y = (long) latitude_GPS - (long) latitude_station ;
       coord_z = altitude_GPS - altitude_station ;

       angle_drone_azimut = angle_azimut_du_drone(coord_x,coord_y) ;
       angle_moteur_azimut = position_moteur_azimut*2.0*pi/488.0 ;
       if( fabs(angle_moteur_azimut-angle_drone_azimut) > pi ) {
          delta_angle_azimut = 2*pi - fabs(angle_moteur_azimut-angle_drone_azimut) ;
          if (angle_moteur_azimut-angle_drone_azimut>0) sens_rotation_azimut= SENS_HOR ;
          else sens_rotation_azimut = SENS_TRIGO ;
       }
```

Attente d'appui sur le bouton pour initialiser la station (moteurs et coordonnées GPS)

Calcul du delta position entre la station et le drone

Calcul du nombre de pas pour le moteur azimut

```
      else {
         delta_angle_azimut = fabs(angle_moteur_azimut-angle_drone_azimut) ;
         if(angle_moteur_azimut-angle_drone_azimut>0) sens_rotation_azimut=SENS_TRIGO ;
         else sens_rotation_azimut = SENS_HOR ;
      }
      nb_pas_azimut = (int) (0.5+(delta_angle_azimut*488.0/(2.0*pi))) ;

      tracking_azimut(1) ;      --> Déplacement du moteur azimut

      x = (int) (11.11949266*coord_x) ;// en mètres
      y = (int) (11.11949266*coord_y) ;// 111194.92... =  Rayon_Terre * 2pi / 360
      angle_drone_profondeur = angle_profondeur_du_drone(x,y,coord_z) ;
      angle_moteur_profondeur = position_moteur_profondeur*2.0*pi/246.0 ;
      delta_angle_profondeur = fabs(angle_moteur_profondeur-angle_drone_profondeur) ;
      if (angle_moteur_profondeur<angle_drone_profondeur ) sens_rotation_profondeur = SENS_HOR ;
      else sens_rotation_profondeur = SENS_TRIGO ;
      nb_pas_profondeur = (int) (0.5+(delta_angle_profondeur*246.0/(2.0*pi))) ;
                                                      --> Calcul du nombre
                                                          de pas pour le
                                                          moteur profondeur

      tracking_profondeur(5) ;  --> Déplacement du moteur profondeur
   }
}
```

La fonction de déplacement des moteurs :

```
void tracking_azimut(int vitesse) {
   int i = 0 ;

   Azimut_SensRotation_297 = sens_rotation_azimut ;
      --> Choix du sens de rotation du moteur
          (la variable sens_rotation_azimut
          est globale)

   Azimut_Enable_297 = 1 ;

   for(i=0;i<nb_pas_azimut;i++) {
      Azimut_Clock_297 = 0;
      Delay10KTCYx(vitesse);
      Azimut_Clock_297 = 1;
      Delay10KTCYx(vitesse);
      if (sens_rotation_azimut==SENS_HOR) position_moteur_azimut ++ ;
      else if (sens_rotation_azimut==SENS_TRIGO) position_moteur_azimut -- ;
   }
      --> Génération d'un signal horloge pour le
          déplacement des moteurs et incrémentation
          de la variable de position du moteur.
}
```

La gestion des interruptions :

```
void InterruptHandler(){

   if ((PIR1bits.RCIF)&&(MAJ_Trame==1)) {   --> On détecte @

      if(RCREG=='@') compteur_octet = -1 ;

      else if (RCREG=='#') {

         if (compteur_octet==-1) compteur_octet=0;
            --> À ce stade là, on a détecté
                consécutivement @ et #.
                On pourra donc inscrire le
                reste de la trame dans le
                tableau trame[]

         else compteur_octet=-2;
            --> Si # n'a pas été détecté,
                on attendra de recevoir à nouveau @
      }

      else if ((compteur_octet>=0)&&(compteur_octet<18)){
         trame[compteur_octet]=RCREG ;
         compteur_octet++;
         if(compteur_octet>=18) {
            MAJ_Trame = 2 ;
            compteur_octet = -2 ;
            RCSTAbits.CREN = 0;
         }
      }
         --> Ici, on récupère la trame et
             on la stocke dans le tableau
             trame[].
             Une fois que la trame est
             complètement chargée, on
             désactive les interruptions
             sur USART

   }
}
```

L'acquisition des données GPS :

```
void acquisition_GPS() {
   unsigned long degre ;
   unsigned long minute ;
   unsigned long datarecue ;

   MAJ_Trame = 1 ;
   RCSTAbits.CREN = 1;
   while(MAJ_Trame != 2) ;

   datarecue = (((unsigned long)trame[0])<<24)
               + (((unsigned long)trame[1])<<16)
               + (((unsigned long)trame[2])<<8)
               + (((unsigned long)trame[3])) ;
   degre = (datarecue/1000000)*1000000 ;
   longitude_GPS = (degre + ((datarecue - degre)/60)*100)/100 ;

   datarecue = (((unsigned long)trame[4])<<24)
               + (((unsigned long)trame[5])<<16)
               + (((unsigned long)trame[6])<<8)
               + (((unsigned long)trame[7])) ;
   degre = (datarecue/1000000)*1000000 ;
   latitude_GPS = (degre + ((datarecue - degre)/60)*100)/100 ;

   altitude_GPS = (trame[12]<<8)+trame[13] ;

   MAJ_Trame = 0 ;
}
```

Ici, on active les interruptions sur USART, et on attend ensuite que la trame soit chargée complètement

Ici, on traite les données de longitude/latitude pour la mettre en forme sous un nombre en degrés pur :
Ex: la trame reçue est sous la forme ddmmmm (d:degré,m:minute),
et on la met sous forme dddddd ($x10^{-4}$).

L'altitude est déjà sous la bonne forme.

La conversion des delta position drone-station en angle (seul le code de l'azimut apparaît ici) :

```
float angle_azimut_du_drone(long coordonnee_x,long coordonnee_y) {
   float angle = 0 ;
   if(coordonnee_y==0) {
      if(coordonnee_x==0) {}
      else if(coordonnee_x>0) angle = pi/2.0 ;
      else if(coordonnee_x<0) angle = -pi/2.0 ;
   }
   else {
      if(coordonnee_x==0) {
         if(coordonnee_y>0) angle = 0 ;
         else angle = pi ;
      }
      else {
         angle = atan(((float)coordonnee_x)/((float)coordonnee_y)) ;
         if(coordonnee_y<0){
            if (coordonnee_x<0) angle = angle - pi ;
            else angle = angle + pi ;
         }
      }
   }
   return angle ;
}
```

On calcule la valeur de l'angle de]-pi;+pi°], tout en prenant en compte les différents cadrans de la fonction arctan()

2) PIC échangeur :

Voici un rappel du rôle de ce PIC. Tout d'abord, il réceptionne en continu la trame GPS. Il va ensuite récupérer par UART une trame PID depuis le PC (via un port USB). Il récupère également les consignes Joystick. Il envoie ensuite une trame unique contenant les PIDs ainsi que les consignes Joystick par Xbee au drone.

Par ailleurs, il fait la réception de la trame GPS pour pouvoir afficher (selon les consignes du PC) des informations sur la lunette 3D.

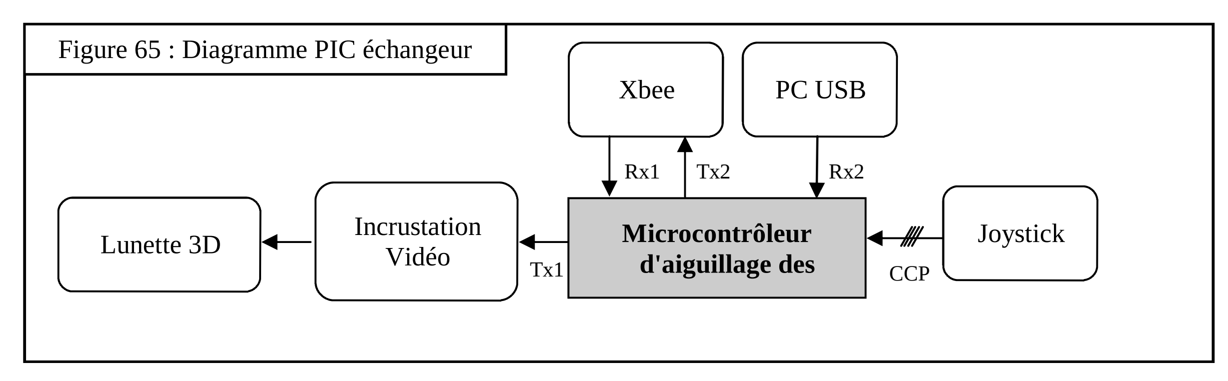


Figure 65 : Diagramme PIC échangeur

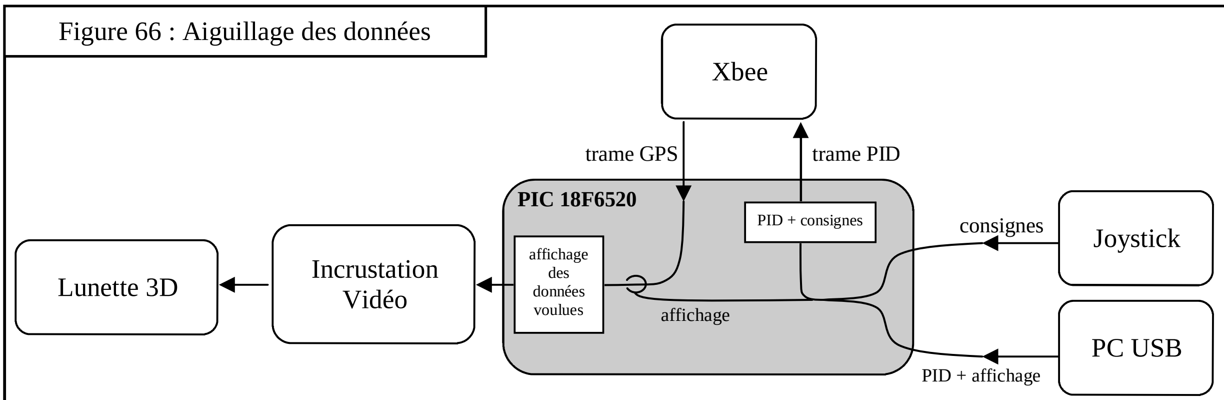


Figure 66 : Aiguillage des données

Idées pour le code :

Le code de ce PIC a été commencé partiellement par Emmanuel Roussel (GE4), des idées pour la personne qui s'en occupera sont également données à titre indicatif dans la suite. Elles ne sont peut être pas réalisables mais ça permettra de refléter ce qui a été pensé lors de la réalisation de la carte :

Pour le choix de l'affichage des données, au début (une fois la station sol démarrée), une trame PID issue du PC pourra par exemple envoyer un code bien précis indiquant si oui ou non telle information devra être affichée.
Suite à cela, le PC n'aura plus besoin d'envoyer ce qu'il faut afficher. En interne, le PIC filtrera les données issue du Xbee à afficher sur la surimpression.

On pourrait également concaténer la consigne Joystick avec les corrections USB, ou bien créer deux trames différentes envoyées tour à tour. (Tout dépend de ce qui est préférable pour la carte mère embarquée).

VI - Mode d'emploi de la station sol V2 :

L'alimentation de la carte se fait par une batterie 12V qui peut fournir suffisamment de courant (des tests sur des alimentations stabilisées de labo n'ont pas été concluant puisque les moteurs consommaient beaucoup trop pour celles-ci).

Brancher le Xbee et vérifier la configuration du Xbee;

Au début, seule la LED 1 s'allumera.

Appuyer sur le bouton INIT.

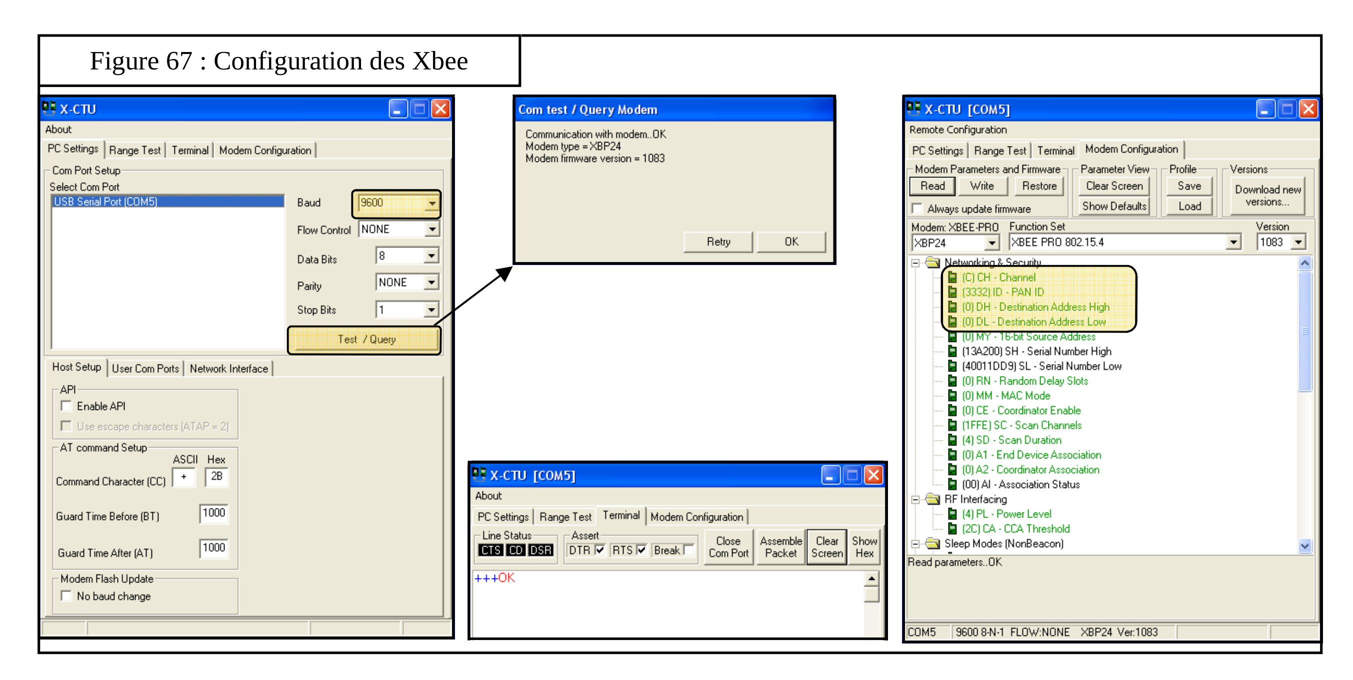
Une fois que le drone est démarré et que la carte capteur du drone envoie la trame GPS correctement, la LED 2 s'allumera.

Commence alors la routine de tracking. La LED 3 clignotera à chaque actualisation.

Configurer le Xbee :

Lorsque le Xbee est branché, et que l'interrupteur placé à côté est calé sur la position "Xbee", on peut brancher un câble USB sur le connecteur USB de la carte (c'est le même câble que ceux des PICKit2). Lors de la première installation, l'ordinateur détectera un composant "FT232RL". Il faudra installer son driver. Il faut ensuite lancer le logiciel X-Ctu.

Figure 67 : Configuration des Xbee



Sur la Figure 67, plusieurs Impressions écran ont été prises avec les paramètres à configurer pour les Xbee :

- Vérifier la communication Modem avec "Test/Query"
- Vérifier dans la bonne communication du Xbee en tapant "+++" sans rien d'autre (ne pas appuyer sur "Entrée" non plus). Un message "OK" devrait apparaitre à côté de +++.
- Si cette communication ne fonctionne pas, modifier les paramètres du Xbee : Channel C, ID (3332), DH (0), DL (0), ainsi que le Baudrate (9600).

Le Xbee est maintenant opérationnel.

VII - Evolutions futures :

La carte station sol V2 n'a pas pu être validée. Il faudrait donc pouvoir vérifier son bon fonctionnement. Avant de continuer à modifier la carte et le code, il serait très intéressant de pouvoir valider celle-ci entièrement. En effet, bien que certains tests aient été concluants, il faudrait poursuivre cette série de test :

Tester toutes les communications :

Xbee → PIC moteur = Normalement Validée
Xbee ↔ PIC aiguilleur = Fonctionnait à un moment (avant modifications Figure 13 et 14)
Xbee ↔ PC USB (switch sur Xbee) = Validée
PIC Aiguilleur → PC USB (switch sur Rx2) = Non Testée
PC USB → PIC Aiguilleur = Fonctionnait avant modifications
Aiguilleur → Incrustation Vidéo = Non Testée

Une fois toutes les communications vérifiées, il faudra peut être songer à reposer une diode faible chute de tension (cf. I-2)2-b) et vérifier si les communications fonctionnent toujours.

Concernant le code en lui-même, celui-ci n'a jamais été testé entièrement sur la V2 car n'ayant plus de disponibilités de la trame GPS. Sur la carte V1, celui-ci a fonctionné assez bien. Par la suite, sur la V2, un code a été fait pour faire tourner les moteurs et ce bout de code a été validé. Si la réception de trame GPS fonctionne (communication Xbee → PIC Moteur), il n'y a pas de raisons pour que ce code ne fonctionne pas.

Pour ce qui est de la carte puissance, celle-ci a été validée. En imposant des signaux Clock (GBF), Enable, Sens, … sur le connecteur DIL10, les moteurs ont tourné comme il fallait (cf. les Figures 59 et 60 des mesures effectuées). Cette carte est donc validée. Des radiateurs non dimensionnés ont été posés sur les hacheurs et le régulateur 5V (cf. Figure 68). Il faudra certainement en dimensionner pour les futures versions.

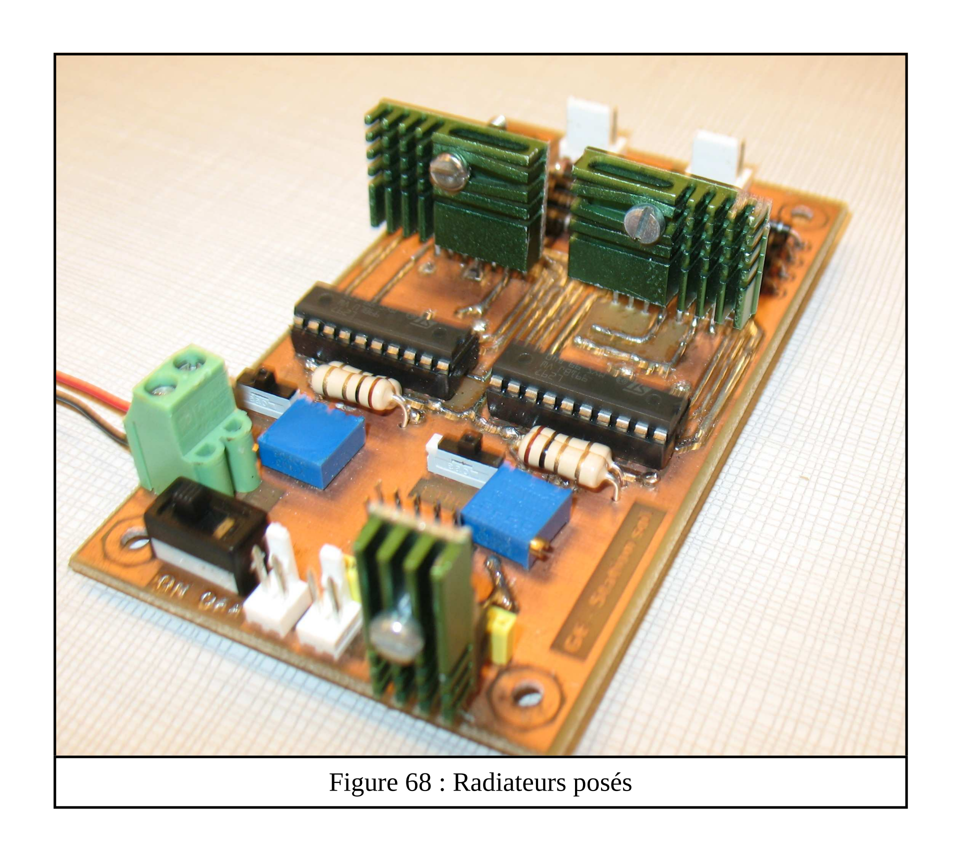


Figure 68 : Radiateurs posés

Concernant d’autres évolutions possibles, il serait intéressant d’interfacer la boussole et ainsi d’éviter une phase d’initialisation de la station (ce n’est cependant pas vraiment une urgence). Il serait bien de stocker également les coordonnées GPS de la station sol dans l’eeeprom de façons à pouvoir rebooter la carte si celle-ci est perturbée. Au redémarrage, même si le drone est encore en vol, la station sol pourra s’initialiser sur cette valeur stockée.

Lorsque la carte commande sera validée (en particulier les communications), un tirage professionnel de la carte pourra être envisagé.

En attendant que les quelques modifications et tests soient effectués sur cette carte, je me tiens à disposition de tout élève voulant faire évoluer la station sol.

Conclusion

J'ai effectué mon stage sur le projet drone de l'INSA. Mon rôle dans ce projet était de développer une station sol permettant un suivi radar du drone grâce à la différence entre les coordonnées GPS de la station et celle du drone. Malgré des recherches actives de stage dans des entreprises, la majorité des entreprises avançait souvent que la crise rendait difficile toute proposition de stage. Le stage effectué à l'INSA a tout de même était très intéressant. Je pense même qu'il aurait été très difficile de trouver un stage technicien de cette qualité en entreprise. Cependant, je n'ai pas pu développer ma culture d'entreprise, ou de manière générale l'aspect organisationnel d'une entreprise.

Le travail que j'ai effectué correspondait à la mise en application d'un grand nombre de connaissances théoriques inculquées par ma formation. Concrètement, j'ai pu travaillé à la fois sur du software (développement de programmes) et sur du hardware (réalisation de cartes électroniques, tests de circuit électronique). Je regrette cependant d'avoir passé trop de temps sur des aspects programmation. En effet, je n'ai donc pas pu m'attarder de manière efficace sur le reste de mon activité et ceci a été très embêtant car la carte électronique n'a pas pu être validée entièrement. J'avais donc initialement mal prévu mon diagramme de Gantt.

Concernant l'aspect projet, j'ai trouvé la façon de travailler très agréable. Tout d'abord, l'ambiance au sein de l'équipe de travail était particulièrement amicale. Il était possible de se rattacher ou demander conseil à quelqu'un d'autre. Pendant les deux mois du stage, le travail a été intensif, avec assez régulièrement des heures de travail en plus. L'aspect essais sur le terrain m'a également montré l'importance d'avoir des périodes de tests pour valider un maximum de points.

Durant ce stage, j'ai également découvert le monde de l'aéromodélisme. Travailler sur un projet aussi innovant a aussi été très enrichissant. En effet, ce stage m'a donné une vision bien plus précise de ce monde et en particulier des problèmes auxquels peuvent être confrontés les créateurs de tels engins. J'ai par ailleurs eu l'occasion de côtoyer le milieu des drones au plus près lors de la conférence sur les UAV (EMAV 2009) qui s'est tenue à Delft.

Remerciements:

Je souhaiterai remercier tout particulièrement Mr Renaud Kiefer qui a rendu ce stage possible, mais également mes camarades de projet qui ont contribué à une très bonne ambiance au sein de l'équipe de projet